Data General
Data General Corporation, Westboro, Massachusetts 01580

Customer Documentation

System Manager's Reference for the DG/UX™ System

093701050-03

AViiON®
PRODUCT LINE

System Manager's Reference for the DG/UX™ System

093-701050-03

For the latest enhancements, cautions, documentation changes, and other information on this product, please see the Release Notice (085-series) supplied with the software.

Ordering No. 093-701050

Printed in the United States of America
Revision 03, February 1992

NOTICE

System Manager's Reference for the DG/UX System

093-701050-03

Revision History:	Effective with:
Original Release – February 1990	DG/UX 4.20
Revision 1 – June 1990	DG/UX 4.30
Revision 2 – June 1991	DG/UX 5.4
Revision 3 – February 1992	DG/UX 5.4.1

Preface

This *System Manager's Reference for the DG/UX™ System* describes the commands, file formats, system special files, and system maintenance procedures used by those who administer a DG/UX system running on an AViiON® computer.

This manual is part of a five-volume reference set. The other manuals are the *User's Reference for the DG/UX System* and the three-volume *Programmer's Reference for the DG/UX System*. These manuals contain in printed (typeset) form the online entries released with the DG/UX System in **/usr/catman** for access by the **man** command.

For step by step instructions on system management, see *Installing the DG/UX™ System*, *Customizing the DG/UX™ System*, and *Managing the DG/UX™ System*. Other related manuals are listed under "Related Manuals" at the end of this manual.

Man Pages

For historical reasons, each entry is called a "manual page" or "man page," though an entry may occupy more than one physical page and may contain more than one entry. If the man page contains more than one entry, it is alphabetized under its "primary" name; for example, the **accept** manual page describes the **accept** and **reject** commands.

Manual pages are assigned to classes ranging from 0 through 8 for easy cross-reference. The class number appears in parentheses following the name; for example, in **accept**(1M) the "1" indicates that **accept** is a command, and the "M" indicates that the man page is in the *System Manager's Reference*.

A command followed by a (1) or (1G) usually means that it is described in the *User's Reference*. (Class 1 commands appropriate for use by programmers are located in the *Programmer's Reference*.) A man page name with a (1M), (4M), (7), or (8) following it means that the entry is in the *System Manager's Reference*. Names with (2) or (3*x*), (4), (5) [except **editread**(5)], or (6F) are in the *Programmer's Reference*. Occasionally, DG/UX man pages refer to other products' man pages, which are not part of the DG/UX documentation; these are so noted.

Manual Organization

The *System Manager's Reference* has four chapters:

Chapter 1: System Maintenance Commands and Application Programs (1M)
This chapter contains commands and programs that are used in administering a DG/UX system.

Chapter 2: File Formats (4M)
This chapter documents the structure of particular system management files.

Chapter 3: Special Files (7)
This chapter discusses the characteristics of system files that refer to input/output devices.

Chapter 4: System Maintenance Procedures (8)
This chapter discusses crash recovery and procedures to remake the system.

Appendix A: Contents and Permuted Index Man Pages
These manual pages contain information extracted from the DG/UX man pages in all five reference volumes.

Man Page Format

Each man page has at least some of the following sections:

NAME	gives the primary name (and secondary names, as the case may be) and briefly states its purpose.
SYNOPSIS	summarizes the usage of the program being described.
DESCRIPTION	discusses how to use these commands.
EXAMPLES	gives examples of usage, where appropriate.
FILES	contains the file names that are referenced by the program.
EXIT CODES	discusses values set when the command terminates. The value set is available in the shell environment variable "?" (see **sh**(1)).
DIAGNOSTICS	discusses the error messages that may be produced. Messages that are intended to be self-explanatory are not listed.
SEE ALSO	offers pointers to related information.
NOTES	gives information that may be helpful under the particular circumstances described.

Some man pages may contain other heads such as **ENVIRONMENT** and **CAVEATS**.

Man Page Notation Conventions

This manual uses certain symbols and styles of type to indicate different meanings in man pages. Those symbol and typeface conventions are defined in the following list. You should familiarize yourself with these conventions before reading the manual.

The description of convention meanings uses the terms "command line," "format line," and "syntax line." A command line is an example of a command string that you should type verbatim; it is preceded by a system prompt. A format line shows how to structure a command; it shows the variables that must be supplied and the available options. A syntax line is a fragment of program code that shows how to use a particular routine; some syntax lines contain variables.

Convention	Meaning
boldface	This font is used for section heads and subsection heads. It is also used to distinguish input from output in examples where the two are intermixed.
`constant width/ monospace`	In command formats and code syntax: This typeface indicates text (including punctuation) that you type verbatim from your keyboard. In text: This typeface is used for examples, code samples, pathnames, and the names of commands, files, directories, and manual pages. In all contexts: The following characters, which have special meanings explained below, do not have special meaning but simply represent themselves when they appear in constant-width font: `< > [ ] { } |`. In constant-width font they are are I/O redirection operators, brackets, braces, and the pipe symbol.
italic	In format lines: This font represents variables for which you supply values; for example, the names of your directories and files, your username and password, and possible arguments to commands.
[*optional*]	In format lines: Regular-font brackets surround an optional argument. Don't type the brackets; they only set off what is optional. These brackets should not be confused with constant-width brackets.
choice1|*choice2*	In format lines: The vertical bar indicates a choice between *choice1* and *choice2*.
...	In format lines and syntax lines: You can repeat the preceding argument as many times as desired.
{ }	In format lines: These regular-font braces surround either two or more choices or syntax elements that are repeatable as a group.
< >	In command lines and other examples: Angle brackets distinguish a command sequence or a keystroke (such as **<Ctrl-D>**, **<Esc>**, and **<3dw>**) from surrounding text. Note that these angle brackets are in regular type and that you do not type them; there are, however, constant-width versions of these symbols that you do type.
`$, %, #`	In command lines and other examples: These symbols represent the system command prompt symbols used for the Bourne and Korn shells, the C shell, and the superuser, respectively. Note that your system might use different symbols for the command prompts.

Contacting Data General

Data General wants to assist you in any way it can to help you use its products. Please feel free to contact the company as outlined below.

Manuals

If you require additional manuals, please use the enclosed TIPS order form (United States only) or contact your local Data General sales representative. A list of related documents appears at the end of this manual with the TIPS order form.

For a complete list of AViiON® and DG/UX™ manuals, see the *Guide to AViiON® and DG/UX™ System Documentation* (069-701085). The on-line version of this manual found in **/usr/release/doc_guide** contains the most current list.

Telephone Assistance

If you are unable to solve a problem using any manual you received with your system, free telephone assistance is available with your hardware warranty and with most Data General software service options. If you are within the United States or Canada, contact the Data General Customer Support Center (CSC) by calling 1-800-DG-HELPS. Lines are open from 8:00 a.m. to 5:00 p.m., your time, Monday through Friday. The center will put you in touch with a member of Data General's telephone assistance staff who can answer your questions.

For telephone assistance outside the United States or Canada, ask your Data General sales representative for the appropriate telephone number.

Joining Our Users Group

Please consider joining the largest independent organization of Data General users, the North American Data General Users Group (NADGUG). In addition to making valuable contacts, members receive FOCUS monthly magazine, a conference discount, access to the Software Library and Electronic Bulletin Board, an annual Member Directory, Regional and Special Interest Groups, and much more. For more information about membership in the North American Data General Users Group, call 1-800-932-6663 or 1-508-443-3330.

End of Preface

Contents

Chapter 1 — Administrative Commands and Application Programs

Tables

Chapter 1
Administrative Commands and Application Programs

This chapter contains reference entries describing DG/UX, TCP/IP,and NFS commands used chiefly for system maintenance and administration. These manual pages are also supplied on the product release tape and can be accessed online via the **man** command. Table 1-1 summarizes the TCP/IP administrative commands.

Table 1-1 Summary of TCP/IP Administrative Commands

Name	Description
arp(1M)	Address resolution and display
dbm(1M)	General database management tool
ftpd(1M)	File Transport Protocol server
getmany(1M)	Get Management Information Base (MIB) objects
getnext(1M)	Get next Management Information Base (MIB) object
getone(1M)	Get a Management Information Base (MIB) object
ifconfig(1M)	Configure DG/UX system network interface
inetd(1M)	Internet services server
initrarp(1M)	Initialize ARP table through RARP
mailstats(1M)	Print sendmail statistics
named(1M)	Domain name server process
netinit(1M)	Build TCP/IP protocol stack
netstat(1M)	Show status for DG/UX network parameters
nslookup(1M)	Query domain name servers
ping(1M)	Network debugging
pmtd(1M)	Magnetic tape pseudo device server
rexecd(1M)	Remote execution server
rlogind(1M)	Remote login server
route(1M)	Manipulate the routing tables
routed(1M)	Network routing server
rshd(1M)	Remote shell server
rwhod(1M)	System status server
sendmail(1M)	Internet mail transport service
setany(1M)	Set Management Information Base (MIB) object
snmpd(1M)	Simple Network Management Protocol (SNMP) server
telnetd(1M)	TELNET protocol server
tftpd(1M)	TFTP protocol server
trap_recv(1M)	Receive SNMP traps
trap_send(1M)	Send SNMP traps

Table 1-2 summarizes the NFS administrative commands.

Table 1-2 Summary of NFS Administrative Commands

Name	Description
automount(1M)	Automatically mount NFS file systems.
bootparamd(1M)	Boot parameter server.
exportfs(1M)	Export and unexport directories to NFS clients.
keyserv(1M)	Server for storing public and private keys.
lockd(1M)	Network lock server
makedbm(1M)	Make an NIS **dbm** file.
mountd(1M)	NFS mount request server.
newkey(1M)	Create a new key in the public key database.
nfsd(1M)	Network File System servers
nfsstat(1M)	Display Network File System statistics.
portmap(1M)	DARPA port to RPC program number mapper.
rexd(1M)	RPC-based remote execution server.
rpcinfo(1M)	Report RPC information.
rstatd(1M)	Return performance statistics from kernel.
rusersd(1M)	Network username server.
rwalld(1M)	Network **rwall** server.
showmount(1M)	Show all remote mounts.
spray(1M)	Spray packets.
sprayd(1M)	Spray server.
statd(1M)	Network status monitor.
xdrtoc(1M)	Convert distribution table of contents to ASCII.
ypinit(1M)	Build and install the NIS database.
ypmake(1M)	Rebuild the NIS database.
yppasswdd(1M)	Server for modifying the NIS password file.
yppoll(1M)	Show what version of NIS map is at NIS server host.
yppush(1M)	Force propagation of a changed NIS map.
ypserv(1M)	NIS server and bind processes.
ypset(1M)	Point **ypbind** at a particular server.
ypupdated(1M)	Server for changing NIS information.
ypxfr(1M)	Transfer NIS map from an NIS server to here.

The man pages for user commands and programmer commands are found in Chapter 1 of the *User's Reference for the DG/UX System* and Chapter 1 of the *Programmer's Reference for the DG/UX System (Volume 1)*.

NAME

`intro` – introduction to system maintenance commands and application programs

DESCRIPTION

This chapter describes, in alphabetical order, commands that are used chiefly for system maintenance and administration. Other DG/UX commands appear in Chapter 1 of the *User's Reference for the DG/UX System* and Chapter 1 of the *Programmer's Reference for the DG/UX System (Volume 1)*. See the `contents`(0) manual page for an alphabetized list of all DG/UX commands.

Command Syntax

Unless otherwise noted, commands described in this chapter accept options and other arguments according to the following syntax:

name [*option*(*s*)] [*cmdarg*(*s*)]

name	The name of an executable file.
option	– *noargletter*(*s*) or, – *argletter*<>*optarg* where <> is optional white space.
noargletter	A single letter representing an option without an argument.
argletter	A single letter representing an option requiring an argument.
optarg	Argument (character string) satisfying preceding *argletter*.
cmdarg	Pathname (or other command argument) *not* beginning with –, or – by itself, indicating the standard input.

DIAGNOSTICS

Upon termination, each command returns two bytes of status, one supplied by the system and giving the cause for termination, and (in the case of normal termination) one supplied by the program (see `wait`(2) and `exit`(2)). The former byte is 0 for normal termination. The latter is customarily 0 for successful execution, and nonzero for such problems as erroneous parameters, or bad or inaccessible data. It is called variously "exit code", "exit status", or "return code", and is described only where special conventions are involved.

SEE ALSO

`contents`(0), `getopt`(1), `getopt`(3C).

NOTE

Many commands do not adhere to the described syntax.

NAME

accept, reject – accept or reject print requests

SYNOPSIS

accept *destinations*
reject [-r *reason*] *destinations*

DESCRIPTION

accept allows the queueing of print requests for the named *destinations*. A *destination* can be either a printer or a class of printers. Run lpstat -a to find the status of *destinations*.

reject prevents queueing of print requests for the named *destinations*. A *destination* can be either a printer or a class of printers. (Run lpstat -a to find the status of *destinations*.) The following option is useful with reject.

-r *reason* Assign a *reason* for rejection of requests. This *reason* applies to all *destinations* specified. *Reason* is reported by lpstat -a. It must be enclosed in quotes if it contains blanks. The default reason is unknown reason for existing destinations, and new destination for destinations just added to the system but not yet accepting requests.

FILES

/var/spool/lp/*

SEE ALSO

lpadmin(1M), lpsched(1M).
enable(1), lp(1), lpstat(1) in the *User's Reference Manual*.

NAME

acctdisk, acctdusg, accton, acctwtmp - overview of accounting and miscellaneous accounting commands

SYNOPSIS

/usr/lib/acct/acctdisk

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

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

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

DESCRIPTION

Accounting software is structured as a set of tools (consisting of both C programs and shell procedures) that can be used to build accounting systems. Acctsh(1M) describes the set of shell procedures built on top of the C programs.

Connect time accounting is handled by various programs that write records into /etc/utmp, as described in utmp(4). The programs described in acctcon(1M) convert this file into session and charging records, which are then summarized by acctmerg(1M).

Process accounting is performed by the DG/UX system kernel. Upon termination of a process, one record per process is written to a file (normally /usr/adm/pacct). The programs in acctprc(1M) summarize this data for charging purposes; acctcms(1M) is used to summarize command usage. Current process data may be examined using acctcom(1).

Process accounting and connect time accounting (or any accounting records in the format described in acct(4)) can be merged and summarized into total accounting records by *acctmerg* (see tacct format in acct(4)). *Prtacct* (see acctsh(1M)) is used to format any or all accounting records.

Acctdisk reads lines that contain user ID, login name, and number of disk blocks and converts them to total accounting records that can be merged with other accounting records.

Acctdusg reads its standard input (usually from find / -print) and computes disk resource consumption (including indirect blocks) by login. If -u is given, records consisting of those file names for which *acctdusg* charges no one are placed in *file* (a potential source for finding users trying to avoid disk charges). If -p is given, *file* is the name of the password file. This option is not needed if the password file is /etc/passwd. (See diskusg(1M) for more details.)

Accton alone turns process accounting off. If *file* is given, it must be the name of an existing file, to which the kernel appends process accounting records (see acct(2) and acct(4)).

Acctwtmp writes a utmp(4) record to its standard output. The record contains the current time and a string of characters that describe the *reason*. A record type of ACCOUNTING is assigned (see utmp(4)). *Reason* must be a string of 11 or less characters, numbers, $, or spaces. For example, the following are suggestions for use in reboot and shutdown procedures, respectively:

```
acctwtmp "uname" >> /etc/wtmp
acctwtmp "file save" >> /etc/wtmp
```

Note: If you are using NFS, the yellow pages (YP) will be used to convert user IDs to login names for those users not listed in /etc/passwd.

FILES

/etc/passwd	used for login name to user ID conversions
/usr/lib/acct	holds all accounting commands listed in Chapter 1M of this manual
/usr/adm/pacct	current process accounting file
/etc/wtmp	login/logoff history file

SEE ALSO

`acctcms`(1M), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `diskusg`(1M), `fwtmp`(1M), `runacct`(1M), `acctcom`(1), `acct`(2), `acct`(4), `utmp`(4).

Installing the DG/UX System, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NAME

acctcms - command summary from per-process accounting records

SYNOPSIS

/usr/lib/acct/acctcms [-a [-po]] [-cjnst] *files*

DESCRIPTION

Acctcms reads one or more *files*, normally in the form described in acct(4). It adds all records for processes that executed identically-named commands, sorts them, and writes them to the standard output, normally using an internal summary format. Options are:

-a Print output in ASCII rather than in the internal summary format. The output includes command name, number of times executed, total kcore-minutes, total CPU minutes, total real minutes, mean size (in kilobytes), mean CPU minutes per invocation, "hog factor" (a number from .00-.99 indicating the amount of system memory and time used), characters transferred, and blocks read and written, as in acctcom(1). Output is normally sorted by total kcore-minutes.

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

-j Combine all commands invoked only once under "***other".

-n Sort by number of command invocations.

-s Any file names encountered afterward are already in internal summary format.

-t Process all records as total accounting records. The default internal summary format splits each field into prime and non-prime time parts. This option combines the prime and non-prime time parts into a single field that is the total of both, and provides upward compatibility with old (i.e., UNIX® System V) style acctcms internal summary format records.

-p Output a prime-time-only command summary.

-o Output a non-prime-time-only (offshift) command summary.

When -p and -o are used together, a combination prime and non-prime time report is produced. All the output summaries will be total usage except number of times executed, CPU minutes, and real minutes, which will be split into prime and non-prime.

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

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

FILES

/usr/lib/acct/holidays Defines non-prime hours and days

SEE ALSO

acct(1M), acctcon(1M), acctmerg(1M), acctprc(1M), acctsh(1M), fwtmp(1M), runacct(1M), acctcom(1), acct(2), acct(4), holidays(4), utmp(4).

NOTE

Unpredictable output results if -t is used on new style internal summary format files or if it is not used with old style internal summary format files.

NAME

acctcon1, acctcon2 - connect-time accounting

SYNOPSIS

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

/usr/lib/acct/acctcon2

DESCRIPTION

Acctcon1 converts a sequence of login/logoff records read from its standard input to a sequence of records, one per login session. This command's input should normally be redirected from /etc/wtmp. Its output is ASCII, giving device, user ID, login name, prime connect time (seconds), non-prime connect time (seconds), session starting time (numeric), and starting date and time. Options are:

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

-t Acctcon1 maintains a list of lines on which users are logged in. When it reaches the end of its input, it emits a session record for each line that still appears to be active. It normally assumes that its input is a current file, so that it uses the current time as the ending time for each session still in progress. The -t flag causes it to use, instead, the last time found in its input, thus assuring reasonable and repeatable numbers for non-current files.

-l Create *file* to contain a summary of line usage showing line name, number of minutes used, percentage of total elapsed time used, number of sessions charged, number of logins, and number of logoffs. This file helps track line usage, identify bad lines, and find software and hardware oddities. Hang-up, termination of login(1) and termination of the login shell each generate logoff records, so that the number of logoffs is often three to four times the number of sessions. See init(1M) and utmp(4).

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

Acctcon2 expects as input a sequence of login session records and converts them into total accounting records (see tacct format in acct(4)).

EXAMPLES

These commands are typically used as shown below. The file ctmp is created only for the use of acctprc(1M) commands:

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

FILES

/etc/wtmp

/usr/lib/acct/holidays Defines non-prime hours and days

SEE ALSO

acct(1M), acctcms(1M), acctcom(1), acctmerg(1M), acctprc(1M), acctsh(1M), fwtmp(1M), init(1M), login(1), runacct(1M), acct(2), acct(4), holidays(4), utmp(4).

NOTE

The line usage report is confused by date changes. Use wtmpfix (see fwtmp(1M)) to correct this situation.

NAME

acctmerg - merge or add total accounting files

SYNOPSIS

/usr/lib/acct/acctmerg [-aiptuv] [*file*] ...

DESCRIPTION

Acctmerg reads its standard input and up to nine additional files, all in the tacct format (see acct(4)) or an ASCII version thereof. It merges these inputs by adding records whose keys (normally user ID and name) are identical, and expects the inputs to be sorted on those keys. Options are:

- -a Produce output in ASCII version of tacct.
- -i Input files are in ASCII version of tacct.
- -p Print input with no processing.
- -t Produce a single record that totals all input.
- -u Summarize by user ID, rather than user ID and name.
- -v Produce output in verbose ASCII format, with more precise notation for floating point numbers.

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

EXAMPLES

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

SEE ALSO

acct(1M), acctcms(1M), acctcom(1), acctcon(1M), acctprc(1M), acctsh(1M), fwtmp(1M), runacct(1M), acct(2), acct(4), utmp(4).

NAME

acctprc1, acctprc2 – process accounting

SYNOPSIS

/usr/lib/acct/acctprc1 [*ctmp*]

/usr/lib/acct/acctprc2

where:

ctmp Pathname of a file containing a list of login sessions in the form described in acctcon(1M), sorted by user ID and login name; the default is the password file.

DESCRIPTION

The acctprc1 command reads input in the form described by acct(4) and adds login names corresponding to user ID's. It then writes for each process an ASCII line giving user ID, login name, prime CPU time (ticks), non-prime CPU time (ticks), and mean memory size (in memory segment units). The information in *ctmp* helps it distinguish among different login names that share the same user ID.

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

These commands are typically used as shown below:

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

FILES

/etc/passwd

/usr/lib/acct/holidays Defines non-prime hours and days

SEE ALSO

acct(1M), acctcms(1M), acctcom(1), acctcon(1M), acctmerg(1M), acctsh(1M), cron(1M), fwtmp(1M), runacct(1M), acct(2), acct(4), holidays(4), utmp(4).

NOTE

If you are using the Network File System (NFS™), the yellow pages (YP) will be used to convert user IDs to login names for those users not listed in /etc/passwd.

NAME

chargefee, ckpacct, dodisk, lastlogin, monacct, nulladm, prctmp, prdaily, prtacct, shutacct, startup, turnacct – shell procedures for accounting

SYNOPSIS

/usr/lib/acct/chargefee *login-name number*

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

/usr/lib/acct/dodisk

/usr/lib/acct/lastlogin

/usr/lib/acct/monacct *number*

/usr/lib/acct/nulladm *file*

/usr/lib/acct/prctmp

/usr/lib/acct/prdaily [*mmdd*]

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

/usr/lib/acct/shutacct ["*reason*"]

/usr/lib/acct/startup

/usr/lib/acct/turnacct on|off|*switch*

DESCRIPTION

You can invoke chargefee to charge a *number* of units to *login-name*. A record is written to /usr/adm/fee, to be merged with other accounting records during the night.

Initiate ckpacct via cron(1M). It periodically checks the size of /usr/adm/pacct. If the size exceeds *blocks*, 1000 by default, turnacct is invoked with argument *switch*. If the number of free disk blocks in the /usr file system falls below 500, ckpacct automatically turns off the collection of process accounting records via the off argument to turnacct. When at least 500 blocks are restored, accounting is reactivated. This feature is sensitive to the frequency at which ckpacct is executed, usually by cron.

Invoke dodisk by cron to perform the disk accounting functions. The disk accounting functions keep track of disk usage by user. By default, disk accounting is performed on *special files* in /etc/fstab.

Lastlogin is invoked by runacct (see runacct(1M)) to update /usr/adm/acct/sum/loginlog, which shows the last date each person logged in.

Invoke monacct once each month or each accounting period. *Number* indicates which month or period it is. If you omit *number*, it defaults to the current month (01–12). This default is useful if monacct is executed via cron(1M) on the first day of each month. monacct creates summary files in /usr/adm/acct/fiscal and restarts summary files in /usr/adm/acct/sum.

Nulladm creates *file* with mode 664 and ensures that owner and group are adm. It is called by accounting shell procedures.

Prctmp prints the session record file (normally /usr/adm/acct/nite/ctmp) created by acctcon(1M).

Prdaily is invoked by runacct (see runacct(1M)) to format a report of the previous day's accounting data. The report resides in /usr/adm/acct/sum/rprt*mmdd*, where *mmdd* is the month and day of the report.

You can print the current daily accounting reports by typing `prdaily`. To print a previous day's accounting reports, use the *mmdd* option and specify the report date.

`Prtacct` formats and prints any total accounting (`tacct`) file.

During a system shutdown, you should invoke `shutacct` to turn process accounting off and append a "reason" record to `/etc/wtmp`.

`Startup` is called by `/usr/sbin/init.d/rc.account` when the system is brought up, if the `account_START` variable has been set correctly. You must set the `account_START` shell variable in `/etc/dgux.params` to "*true*" to have accounting started by `rc.account`.

`Turnacct` is an interface to `accton` (see `acct`(1M)) to turn process accounting `on` or `off`. The *switch* argument turns accounting off, moves the current `/usr/adm/pacct` to the next free name in `/usr/adm/pacct`*incr*, then turns accounting back on again. *Incr* is a number starting with `1` and increasing by one for each additional `pacct` file. Since `ckpacct` calls this procedure, `cron` can maintain it and keep `pacct` to a reasonable size. Accordingly, `acct` starts and stops process accounting via `init` and `shutdown`.

FILES

`/usr/adm/fee`	Accumulator for fees.
`/usr/adm/pacct`	Current file for per-process accounting.
`/usr/adm/pacct*`	Used if pacct gets large and during execution of daily accounting procedure.
`/etc/wtmp`	Login/logoff summary.
`/usr/adm/acct/nite`	Working directory.
`/usr/lib/acct`	Holds all accounting commands listed in Chapter 1M of this manual.
`/usr/adm/acct/sum`	Summary directory, should be saved.

SEE ALSO

`acct`(1M), `acctcms`(1M), `acctcom`(1), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `cron`(1M), `fwtmp`(1M), `runacct`(1M), `acct`(2), `acct`(4), `utmp`(4).

NAME

admaccounting - manage accounting system

SYNOPSIS

```
admaccounting -o start|stop
admaccounting -o list -r { cmd | user } [ -l line-limit ] [ -qv ]
```

DESCRIPTION

The admaccounting command is used to manage system accounting, including starting and stopping system accounting and producing any of several accounting reports.

More information about the accounting system can be obtained from acct(1M) and related manual entries.

Operations

`start` Enable system accounting, if not already enabled.

`stop` Disable system accounting, if enabled.

`list` Display an accounting report.

Options

The following options are allowed for the `list` operation.

`-l` *line-limit*
: Display up to *line-limit* lines of the report. This limit does not include the headers which may precede the report or the totals which follow the report.

`-q`
: Use quiet mode. Headers are not displayed for the `list` operation.

`-r` *report-type*
: An identification of the type of accounting report.

  If the *report-type* is `cmd`, a command usage report is displayed. If the *report-type* is `user`, a report of user logins is displayed.

`-v`
: Use verbose mode. Headers are displayed for the `list` operation. This mode is enabled by default.

EXAMPLES

To start system accounting, use

```
admaccounting -o start
```

To get a report of the 30 most time-consuming commands run on the system, use

```
admaccounting -o list -r cmd -l 30 -v
```

FILES

/etc/log/account.rclock Lock file that indicates whether system accounting is enabled.

DIAGNOSTICS

Warnings

A warning message is printed on an attempt to start the accounting system if it is already started, or to stop it if it is not running.

Errors

The accounting system may produce error indications. See the accounting system documentation for descriptions of error conditions.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

acct(1M), acctcms(1M), acctsh(1M), prtacct(1M), sysadm(1M).

NOTES

Only the super-user may start or stop the accounting system.

The behavior of the `start` and `stop` operations is unreliable if you start or stop accounting by using startup(1M) or turnacct(1M) directly.

NAME

admalias - manage mail alias information in the aliases database

SYNOPSIS

admalias -o add [-y] -m *member-list mailalias*

admalias -o modify [-y] [-n *new-mailalias*] [{ -m *member-list* | -r *member-list* | -a *member-list* }] *mailalias*

admalias -o delete [-y] *mailalias* ...

admalias -o list [-qv] [-y] [*mailalias* ...]

DESCRIPTION

admalias manages the mail alias information in the aliases database file. The aliases(4) file contains an alias name that is associated with a list of one or more mail addresses. The alias name may be substituted for the mail addresses when sending mail using mailx(1) or sendmail(1C).

admalias is normally run by the system administrator on the NIS master machine if the system is running NIS, or on any host if the system is not running NIS.

Operations

add — Add a new alias to the aliases file. If the *mailalias* specified already exists, the operation will not be successful.

modify — Modify currently existing information in the local or NIS aliases database.

delete — Delete the specified alias(es) from the aliases database.

list — List the aliases in the aliases database who match the specified command-line options. If no command-line options are given, then the aliases from the local database will be listed. Any user may execute this operation.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default domain name derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-n *new-mailalias*
Specifies a string of printable characters that is the new alias name for the *mailalias*. It may not contain a colon (:) and must be a unique alias name.

-m *member-list*
Specifies the mail addressees who are to be members of the *mailalias*. This list must be a quoted space or comma-separated list of valid mail addresses (see example).

-a *member-list*
Specifies the mail addressees who are to be added to the list of members associated with the *mailalias*. This list must be a quoted space or comma-separated list of valid mail addresses (see example).

-r *member-list*
Specifies the mail addressees who are to be removed from the list of members associated with the *mailalias*. This list must be a quoted space or comma-separated list of addresses (see example). The list must be addressees who exist as members of the *mailalias*.

`-q` Quiet. The headers are not printed when listing mailalias information.

`-v` Verbose. The headers are printed when listing alias information. This option is enabled by default.

OUTPUT

The `list` operation produces a two-column output. The first column contains the *mailalias* and the second column contains the alias *member-list*.

EXAMPLES

```
admalias -o add -m "root@beast root@jester root@viper" sysadmin
admalias -o modify -a smith,root@parkplace sysadmin
```

DIAGNOSTICS

Warnings

- The `delete` operation is requested and the *mailalias* to delete is not in the aliases database.

Errors

- The `add` operation is requested and *mailalias* already exists.
- The `modify` operation is requested and *mailalias* to modify does not exist.
- The `modify` operation is requested and *new-mailalias* already exists as an alias in the database.

Exit Codes

This section lists the possible exit codes and what they mean.

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

FILES

/etc/aliases

Local aliases database file.

SEE ALSO

`domainname`(1), `mailx`(1), `sendmail`(1C), `sysadm`(1M), `aliases`(4).

NOTES

You must have write permission to the aliases database to use the `add`, `delete`, and `modify` operations. Usually, only the super-user has such permission.

NAME

admbackup - manage backup and recovery of file systems

SYNOPSIS

admbackup -o create [-f *tape*] [-M *medium*] [-n] [-O *dump2-option*] [-p] [*file-system* ...]

admbackup -o list [-qv] [-f *tape*]

admbackup -o restore [-i] [-d *directory*] [-f *tape*] [-M *medium*] *file-system* [*file* ...]

DESCRIPTION

The admbackup command manages the archiving and recovery of file systems.

Backups are typically performed daily, weekly, and monthly. In order to recreate the state of any file or file system, restore (with -o restore) the previous monthly dump, each of the month's weekly dumps in order, and each of the week's daily dumps in order.

A cycle list is maintained by admdumpcycle to keep track of which type of backup (daily, weekly, or monthly) is to be performed next. If at some point during the month the position in the cycle is inaccurate, the position operation of admdumpcycle must be executed to set the current position to the correct entry in the table.

Archives are created with dump2(1M) and restored with restore(1M).

Operations

create Create a backup tape (using dump2(1M)) of all appropriate file systems. The dump cycle list is used to determine the level of the dump being performed today (this level can be overridden by specifying a dump level with the -O *dump2-option* option). If no file systems are given on the command line, or the special keyword all is used, the file system table (see fstab(4)) is used to determine which file systems match the current cycle entry. If the -n option is not used, and no *file-system*s are given on the command line, the current position in the dump cycle list is advanced to the next position.

Backup tapes can be either "packed" or "not packed". Packed tapes contain as many file systems as possible, one after the other. This uses the smallest possible number of tapes for the backup. Backup tapes which are not packed have only one file system per tape. Unpacked tapes minimize risk of lost data if one of the tapes becomes unreadable; it also allows faster recovery of data because there is less data on each tape.

list The list operation with no options lists the file systems which need to be backed up today. In this case, admbackup consults the dump cycle table to determine the current cycle entry, and then searches the file system table to determine which file systems belong to the current cycle entry. If the -f *tape* option is given, the tape in drive *tape* is read to determine which file systems are archived on that tape; these file systems are listed to stdout.

restore Use restore(1M) to restore one or more files or an entire file system from a tape created with -o create. The interactive mode of restore (specified with the -i option) is useful for perusing the directories on the backup tape and extracting individual files. The non-interactive mode is useful for recovering individual files or an entire file system.

If no *files* are given, the entire file system is recovered. Otherwise, only the named *files* are extracted. The *files* must be specified relative to the root of the *file-system*.

Options

`-d` *directory*
: Place the recovered files in *directory*. If this option is not given, the files are placed in the current directory.

`-f` *tape*
: Use *tape* as the tape drive. The default is "/dev/rmt/0".

`-i`
: Use the interactive mode of the `restore`(1M) command. This allows you to move through the directories on the backup tape, selecting which files will be recovered.

`-M` *medium*
: Use *medium* as the medium name. When this option is used with `-o restore`, the *medium* must be the same as for the corresponding `-o create`. The *medium* must be a name from the dump device table (see `dumptab`(4)). The default is "default".

`-n`
: Do not update the database files which keep track of when backups occur. This option is useful when an administrator wants to perform an extra backup without disrupting the normal backup schedule.

`-O` *dump2-option*
: Append *dump2-option* to the list of options passed to the `dump2` program.

`-p`
: Pack as many file systems onto each tape as possible. Otherwise, each file system is put onto a separate tape.

`-q`
: Use quiet mode. Headers for the `list` output are not displayed

`-v`
: Use verbose mode. Headers for the `list` output are displayed. This is the default behavior.

Output

The `list` operation writes its output on stdout. The `restore` operation in interactive mode reads from stdin and writes to stdout. During the `create` operation, output from the `dump2`(1M) command will be written to stdout and to the `/var/adm/log/backup.log` file.

FILES

`/etc/sysadm/dumpcycle`
: The default dump cycle for the system.

`/etc/fstab`
: File system table used to determine which file systems are eligible for `create`.

`/var/adm/log/backup.log`
: Log file for `dump2`(1M)'s output.

DIAGNOSTICS

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

admdumpcycle(1M), admdumpdevice(1M), dump2(1M), restore(1M), sysadm(1M), dumpcycle(4), dumptab(4), fstab(4).

NOTES

This command includes the functions of the osysadm(1M) fsdump, fsrestore, and filerestore commands.

Only the super-user may use the create and restore operations.

NAME

admclient - manage operating system clients

SYNOPSIS

admclient -o add -r *release* [-S *server*] [-b *bootstrap*] [-h *home_directory*] [-k *kernel*] [-s *swap_size*] *hostname* ...

admclient -o delete -r *release* [-p] *hostname* ...

admclient -o list [-qv] [*hostname* ...]

admclient -o modify -r *release* [-b *bootstrap*] *hostname* ...

admclient -o set -r *release* [-S *server*] *hostname* ...

DESCRIPTION

The admclient command is run on the server machine to manage client machines that are served by the server. A client is any machine which gets operating system software from another machine. Disk areas are set aside on the server for use by the clients and configuration files within those areas are initialized for correct use by the clients. Database entries are made on the server to keep track of the state and configuration of each client.

Operations

add — Add an OS client to a release area on the server by creating the client's directory structure, client database table entries, and updating files accessible to the client. A client may be added to more than one release. The client must be added to the hosts database (using admhost(1M)) and the ethers database (using admether(1M)) before being added as a client.

delete — Remove an OS client from a release area on the server by updating the server's database and optionally removing the disk areas allocated for the client.

list — Display information about all or selected OS clients.

modify — Change the bootstrap file used for an OS client.

set — Change the release area in which an OS client will boot. The release area name can be specified as "–none–" to keep a client from booting from any release area.

Options

-b *bootstrap*
The pathname of the bootstrap file. The default bootstrap file is /usr/stand/boot.aviion.

-h *home_directory*
The directory in which users' home directories will be placed. If not the special value "none", the directory will be created on the server and exported to the client.

-k *kernel* The pathname of the kernel file. The default kernel file for the PRIMARY release area is /srv/release/PRIMARY/root/_Kernels/dgux.diskless.

-p Preserve the client's root directory structure when deleting a client.

-q Quiet. Omit header lines from the output of the list operation.

-r *release* The name of the release area. Clients may be associated with more than one release area; this identifies which instance of the client that is to be affected.

`-s` *swap_size*
The size of the swap area to create for the client. The default value is `16m`.

`-S` *server* The name of the server machine as known to the client. This defaults to the default host name as returned by the `hostname`(1) command.

`-v` Verbose. Include header lines in the output of the `list` operation. This option is enabled by default.

OUTPUT

If a particular client name or list of client names is provided to the `list` operation, the output includes the following: client host name, release area name, root directory, swap file, swap size, boot file, and kernel file.

If no hostnames or the keyword "all" is specified, the output includes each client name and release area. Unless the `-q` option is used, header line is produced.

FILES

/srv/release/PRIMARY/root/_Kernels/dgux.diskless	The default kernel file.
/usr/stand/boot.aviion	The default bootstrap file.
/home	The default for the home directory pathname.
/tftpboot	Directory containing bootstrap file links for each client.

DIAGNOSTICS

Warnings

- A database file could not be updated.

Errors

- A prototype file was missing.
- The host name is not recognized.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions. The `add`, `delete`, `modify`, and `set` operations can only be used by the super-user.

3 There was an error in the command line.

SEE ALSO

`admether`(1M), `admhost`(1M), `admrelease`(1M), `sysadm`(1M).

NAME

admdate - manipulate the system date, time and time zone

SYNOPSIS

admdate -o set [-m *month-number*] [-d *day-of-month*] [-H *hour*] [-M *minute*] [-Y *year*] [-Z *time-zone*]

admdate -o get [-qv]

DESCRIPTION

admdate sets or displays the current system date, time, and time zone.

Operations

set — Set the current date, time, or time zone. Restart the cron command to incorporate the new date and time.

get — Display the current date, time, and time zone.

Options

The following options may be used:

-d *day-of-month*
: Use *day-of-month* as the new day of the month (1-31). The default is the current day of the month.

-H *hour* — Use *hour* as the new hour (0-23). The default is the current hour.

-M *minute* — Use *minute* as the new minute (0-59). The default is the current minute.

-m *month-number*
: Use *month-number* as the new month of the year (1-12). The default is the current month.

-q — Use quiet mode. The month, day, hour, minute, year, and time zone are written to standard output.

-v — Use verbose mode. The current date, time, and time zone are written to standard output in the appropriate format for the current locale.

-Y *year* — Use *year* as the new year (including the century). If the *year* does not include a century, the current century will be used. The default is the current year.

-Z *time-zone*
: Set the time zone to *time-zone*. *time-zone* is a string of the form defined in timezone(4). The default is the current time zone.

If the -Z *time-zone* option is given, the /etc/TIMEZONE and /etc/TIMEZONE.csh files are updated to reflect the new *time-zone*. If the *time-zone* is the name of a file found in the /usr/lib/locale/TZ directory, the time zone for the system is set to :*time-zone*.

EXAMPLES

In order to set the date to 8 p.m. on March 2 of 1992, use this command line:

```
admdate -o set -m 3 -d 2 -H 20 -M 0 -Y 1992
```

To change to the Pacific time zone, use this command line:

```
admdate -o set -Z PST8PDT
```

FILES

/etc/TIMEZONE
: Updated to reflect the new time zone.

/etc/TIMEZONE.csh
: Updated to reflect the new time zone.

/usr/lib/locale/TZ
: Directory of compiled time zone databases.

OUTPUT

The `get` operation writes the current date, time, and time zone to the standard output.

DIAGNOSTICS

Warnings

- The new date is the same as the old date.

Errors

- There is an error in the format of the new date or time.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`date`(1), `zic`(1), `cron`(1M), `ctime`(3C), `setlocale`(3C), `timezone`(4).

NOTES

The `set` operation restarts the `cron`(1M) command to incorporate the new date. Other commands may not use the new date until the commands are restarted or the system is rebooted.

This command replaces the `osysadm`(1M) *datetime* command.

Only the super-user may perform the `set` operation.

NAME

admdefault - provide an interface to named default sets

SYNOPSIS

admdefault -o create -O *object* -S *set-name* [-d *directory*]

admdefault -o remove -O *object* -S *set-name* [-d *directory*]

admdefault -o *select* -O *object* -S *set-name* [-d *directory*]

admdefault -o *get* -O *object* [-d *directory*] [-f *file*] [-n] [-S *set-name*] [-qv] [*parameter* ...]

admdefault -o *set* -O *object* -S *set-name* [-d *directory*] [-f *file*] *parameter=value* ...

admdefault -o *list* [-d *directory*] [-f *file*] [-O *object*] [-S *set-name*] [-qv]

DESCRIPTION

The admdefault command manages named default sets.

A default set is a group of several related parameters under a common name. A default set is identified by the *object* whose parameters are grouped and the *set-name* which identifies a particular set of values for the *object*'s parameters. Each of the values in a default set is referenced by its *parameter*. This command allows you to set or retrieve the value of a *parameter* in a default set.

There are two special default sets, system and default. The system default set contains the system-wide default values. These values are shipped with the system. The default default set points to the named default set (either system or one of the user-defined sets) which is used to retrieve values of parameters when no default set is explicitly named. Initially, the default set is the same as the system set. This may be changed with the select operation.

Operations

create — Create a new default set for an *object*. Initially, all parameters in the new set have the same values as the special set system. The values of the parameters may be changed later with the set operation.

remove — Remove a named default set from a particular *object*. Note that the special default sets system and default cannot be removed.

select — Associate the special default set default with a particular named default set. The default set is used to retrieve values when no default set is explicitly named.

get — Retrieve the value of some parameter from a particular *set-name* of an *object*. The value is written to the standard output. If no *set-name* is given, the special set default is used. If the -v option is given, the parameter's value is written in the form *parameter=value*. If no *parameter* is given, all parameters from the set are displayed.

set — Set the value of some parameter from a particular *set-name* of an *object*.

list — Output all known objects, set names, or parameters depending on the options given. If no -O *object* option is given, the list operation outputs all known objects. If no -S *set-name* option is given, the list operation outputs all named default sets for the given *object*. If both -O *object* and -S *set-name* are given, the list of parameters for the indicated *set-name* of the *object* is displayed.

Options

`-d` *directory* Use the named default sets defined under *directory*. The default is `/etc/default/sysadm`.

`-f` *file* Use the default set file named by the pathname *file*. If this option is given, `-d`, `-O`, and `-S` are ignored.

`-n` Get the real name of the indicated default set. This is useful for learning which set is being used as the special `default` default set.

`-O` *object* Use the named default sets for *object*.

`-q` Use quiet mode. No headers are written for the `list` operation.

`-S` *set-name*
Use the named default set *set-name*. For the `get` operation, the default is `system`.

`-v` Use verbose mode. Headers are written for the `list` operation, and output from the `get` operation is in the form *parameter=value*. This mode is enabled by default.

EXAMPLES

To list the named default sets for the "tape" object:

```
admdefault -o list -O tape
```

To get the default value of the "drive" parameter of the "tape" default set:

```
admdefault -o get -q -O tape -S default drive
```

FILES

`/etc/default/sysadm` The default *directory*.

DIAGNOSTICS

Warnings

None.

Errors

- The *directory* does not exist.
- The requested default set does not exist.
- The requested parameter does not exist.

Exit Codes

This section lists the possible exit codes and what they mean.

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

In order to use the `create`, `remove`, or `set` operations, the user must have write permission in the *directory*. For the default *directory*, only the super-user has permission to `create`, `remove`, or `set`.

Named defaults sets are provided for the `client`, `release`, `tape`, and `xterminal` objects.

SEE ALSO

`sysadm`(1M).

NAME

admdumpcycle - manage dump cycle tables

SYNOPSIS

admdumpcycle -o select [*name*]

admdumpcycle -o position { reset | forward [*count*] | backward [*count*] }

admdumpcycle -o list [-qv] [all | current]

DESCRIPTION

admdumpcycle manages the dump cycle tables. The system dump cycle table is used by admbackup(1M) to determine which level of backup is to be performed, and which file systems are to be included in the backup.

See dumpcycle(4) for a complete description of the format of the dump cycle table.

Operations

The admdumpcycle command provides operations for selecting a dump cycle to use, positioning within the current dump cycle, and listing one or more dump cycles.

select
: Select one of the provided dump cycles as the default for the system. The list of possible *name*s is provided by using the list operation with the all keyword. If no *name*s are given, the name of the current cycle is displayed.

position
: Change the current position in the dump cycle file. The reset directive moves the current position to the beginning of the cycle. The forward and backward directives move the current position either forward or backward in the cycle by *count* lines. The default *count* is 1.

list
: Display the dump cycle table. If the special keyword all is given, the names of all available dump cycles are displayed. If the special keyword current is given, only the current line from the dump cycle file is displayed.

Options

The following options may be used:

-q
: Use quiet mode. Do not include a header line for the listing.

-v
: Use verbose mode. Include a header line for the listing. This is the default behavior.

EXAMPLES

To set the dump cycle position to the third dump of the cycle, perform these commands:

```
admdumpcycle -o position reset
admdumpcycle -o position forward 2
```

The following command then verifies that the position is in the correct place:

```
admdumpcycle -o list current
```

FILES

/etc/sysadm/dumpcycle
: The default dump cycle for the system.

/etc/sysadm/dumpcycles/*.proto
: Prototypes of possible dump cycles.

/etc/sysadm/dumpcycles/*.desc
Descriptions of all prototype dump cycles.

OUTPUT

The `select` operation writes the name of the current dump cycle to standard output if no *name*s are given on the command line.

The `list` operation normally writes the current dump cycle to standard output. If the `all` keyword is used, just the names of available dump cycles are written to standard output.

DIAGNOSTICS

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

You must have write permission for the dump cycle file to use the `select` and `position` operations. Usually, only the super-user has such permission.

SEE ALSO

`admbackup`(1M), `dumpcycle`(4).

NAME

admdumpdevice - manage the dump device table

SYNOPSIS

admdumpdevice -o add [-b *block-size*] [-c *capacity*] [-d *description*] *name*

admdumpdevice -o modify [-b *block-size*] [-c *capacity*] [-d *description*] [-n *new-name*] *name*

admdumpdevice -o delete *name* ...

admdumpdevice -o list [-qv] [*name* ...]

DESCRIPTION

admdumpdevice manages the table of dump devices. The entries in the table describe media types which can be used for creating backups with dump2(1M) or admbackup(1M). Each entry contains a medium name, the block size to use for reading and writing, the capacity (in bytes) of the medium, and a description of the medium.

Operations

The admdumpdevice command provides operations for adding, deleting, modifying, or listing entries in the table.

add — Append a new entry to the table.

delete — Remove an existing entry from the table.

modify — Change an existing entry in the table.

list — Display one or more entries from the table. If no *name* is given, or *name* is the special keyword all, all entries from the table are displayed.

Options

The following options may be used:

-b *block-size*
: Use *block-size* as the number of 1024-byte blocks to transfer in each read(2) or write(2) operation to the medium. The default is 16.

-c *capacity*
: Use *capacity* as the total capacity (in bytes) of the medium. The value must be a number, possibly followed by a upper- or lower-case b, k, m, or g to indicate bytes, kilobytes, megabytes, or gigabytes. The default is 150M.

-d *description*
: Use *description* as a text string describing the medium.

-n *new-name*
: Change the name of the medium to *new-name*. The default is *name*.

-q
: Use quiet mode. Headers for the listing are not displayed.

-v
: Use verbose mode. Headers for the listing are displayed. This is the default behavior.

EXAMPLE

To add a 525 Mbyte cartridge tape drive to the media table, use a command line like

```
admdumpdevice -o add -c 525M -d "525 MB Cartridge Tape" cartridge525
```

FILES

`/etc/dumptab` The dump device file.

OUTPUT

The `list` operation displays the medium name, block size, capacity in bytes, and description for each *name* given.

DIAGNOSTICS

Warnings

- An attempt was made to list a non-existent entry.

Errors

- An attempt was made to add an already-existing entry.
- An attempt was made to delete a non-existent entry.
- An attempt was made to modify a non-existent entry.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`admbackup`(1M), `dump2`(1M), `sysadm`(1M), `dumptab`(4).

NOTES

You must have write permission to the dump device file to use the `add`, `delete`, and `modify` operations. Usually, only the super-user has such permission.

NAME

admether - manage ether database

SYNOPSIS

admether -o add [-y] -a *host-ether-address host-name*

admether -o modify [-y] [-n *new-host-name*] [-a *host-ether-address*] *host-name*

admether -o delete [-y] *host-name ...*

admether -o list [-y] [-qv] [-s] [*host-name ...*]

DESCRIPTION

admether manages the local or NIS (YP) ethers database. The ethers database consists of a list of host names and the Ethernet address for each.

Operations

add — Create a new ethers entry.

modify — Change an existing entry to contain a new host name or a new Ethernet address, or both.

delete — Remove an entry from the ethers database.

list — List one or more entries from the ethers database. Information is listed about each of the *host-name*s given; if *host-name* is all, information about all hosts is listed.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. If specified with the add, delete, or modify operations, this option is valid only if the machine on which the command is run is the NIS master. The -y option uses the default domain name derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-a *host-ether-address*
host-ether-address is the Ethernet address of the host being added. If this option is excluded on modify, the current Ethernet address for the host is preserved.

-n *new-host-name*
new-host-name is the new name of the host. Without this option, the host name is not changed.

-q — "Quiet." Produce an unformatted listing. Print no headers, print fields in the order they appear in the database, print each entry on a separate line, and delimit fields within an entry with a single space character.

-s — "Sort." Sort the listing by host name.

-v — "Verbose." Produce a formatted listing with headers and aligned columns. This is the default.

admether expects all Ethernet addresses to be of the form

aa:bb:cc:dd:ee:ff,

where *a*, *b*, *c*, *d*, *e*, and *f* are two-digit hexadecimal numbers between 00 and ff. All 17 characters must be given.

OUTPUT

The `list` operation writes its output to stdout.

The verbose form of the `list` operation outputs the name and Ethernet address for each host. Information is printed in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

ethernet_address host_name

FILES

/etc/ethers

Local ethers database file.

/etc/yp/*yp-domain*/ethers

NIS ethers database file.

DIAGNOSTICS

Warnings

None.

Errors

- The `-y` option is specified for an `add`, `delete`, or `modify` operation and the host is not the NIS master.
- The `add` operation is requested, and *host-name* already exists in the ethers database.
- The `modify` operation is requested, and *new-host-name* already exists in the ethers database.
- The *host-ether-address* is not a valid network address.
- The `-y` option is given, and there is an error exporting NIS maps.
- The `delete` or `modify` operation is requested, and *host-name* does not exist in the ethers database.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

You must have write permission to the ethers database to use the `add`, `delete`, and `modify` operations. Usually, only the super-user has such permission.

SEE ALSO

`domainname`(1), `sysadm`(1M), `ethers`(4).

NAME

admfilesystem - manage file systems

SYNOPSIS

admfilesystem -o create [-p *mkfs-options*] *logical-disk*

admfilesystem -o add -f *fs-source* [-p *fstab-options*] [-d *dump-freq*] [-k *fsck-pass*] [-t *type*] [-e [-P *export-options*]] [-x] *mount-directory*

admfilesystem -o modify [-f *fs-source*] [-p *fstab-options*] [-d *dump-freq*] [-k *fsck-pass*] [-t *type*] [-n *new-mount-dir*] [{ -u | [-e] [-P *export-options*] }] [-x] *mount-directory*

admfilesystem -o delete [-x] *mount-directory* ...

admfilesystem -o mount|unmount|export|unexport *mount-directory* ...

admfilesystem -o list [-qvlrme] [*mount-directory* ...]

admfilesystem -o check [-p *fsck-options*] [*mount-directory*]

DESCRIPTION

admfilesystem provides operations for manipulating entries in the fstab(4) and exports(4) databases as well as a number of other operations commonly needed for the management of file systems.

Operations

create — Create a file system on *logical-disk*.

add — Add a file system to the fstab file and, if -e is specified, to the exports file.

modify — Modify a file system's fstab or exports entry. If -e is specified and there is no entry in exports for *mount-directory*, an entry will be added. If -u (unexportable) is specified, the exports entry for *mount-directory* is deleted (if one exists).

delete — Delete a file system from the fstab file and from the exports file.

mount — Mount a file system. An entry for *mount-directory* must exist in fstab.

unmount — Unmount a file system.

export — Exports a file system making it available for mounting by remote users. An entry for *mount-directory* must exist in exports.

unexport — Unexport a file system making it unavailable for mounting by remote users. This operation does NOT delete the file system's entry from the exports file.

list — List information about file systems to stdout. If neither options nor a *mount-directory* is specified, all file systems from fstab are listed. If *mount-directory* is specified, only information about that file system will be displayed.

check — Check file system and correct inconsistencies. This operation performs the fsck(1M) command for the file system specified by *mount-directory*. If mount-directory is not specified, file systems are checked according to the pass numbers specified in fstab.

Options

Options for the `create` operation:

`-p` *mkfs-options*
: Options to be passed to the `mkfs` command. See `mkfs`(1M) for a list of valid `mkfs` command-line options.

Options for the `add` and `modify` operations:

`-d` *dump-freq*
: Dump frequency entry for `fstab` file. The value for *dump-freq* should be one of {dwmx} (d=daily, w=weekly, m=monthly, x=no dump). The default *dump-freq* value for `add` operations is "d" for dg/ux file systems, "x" for nfs file systems. For `modify` operations, the dump frequency value in `fstab` remains unchanged if `-d` *dump-freq* is not specified.

`-e`
: Exportable. This option indicates that an entry for the file system is to be added to the `exports` file, if one does not already exist.

`-f` *fs-source*
: File system source. For local file systems, name of block special device (e.g. /dev/dsk/usr). For remote file systems, hostname and path on server (e.g. your_host:/usr). For `modify` operations, the file system source in `fstab` remains unchanged if `-f` *fs-source* is not specified.

`-k` *fsck-pass*
: Fsck pass number (a non-negative integer). The default *fsck-pass* value for `add` operations is 1 for dg/ux file systems, 0 for nfs file systems. For `modify` operations, the fsck pass value in `fstab` remains unchanged if `-k` *fsck-pass* is not specified.

`-n` *new-mount-dir*
: New mount directory. Change the mount directory for the file system to *new-mount-dir*.

`-p` *fstab-options*
: Comma-separated list of options which are to be included in the `fstab` entry (e.g. "rw,hard,bg"). If `-p` *fstab-options* is not specified, the default *fstab-options* value will be used. The default *fstab-options* value for `add` operations is "rw" for dg/ux file systems, "rw,hard,bg" for nfs file systems. For `modify` operations, the option list in `fstab` remains unchanged if `-p` *fstab-options* is not specified.

`-P` *export-options*
: Comma-separated list of options for the `exports` file entry (e.g. "access=host1:host2,secure"). An empty option list is the default if `-P` *export-options* is not specified.

`-t` *type*
: Type of file system (e.g. dg/ux, nfs). If this option is not specified on an `add` operation, the default value for *type* is determined from *fs-source*. If *fs-source* contains a colon, *type* defaults to "nfs"; otherwise, *type* defaults to "dg/ux". For `modify` operations, the type in `fstab` remains unchanged if `-t` *type* is not specified.

`-u`
: Unexportable. If this option is specified with the `modify` operation, the `exports` file entry for the file system (if one exists) will be deleted.

Option for the `add`, `modify` and `delete` operations:

`-x`
: Execute immediately. After modifying the appropriate databases, the necessary combination of unexport, unmount, mount, and export

operations are performed to make the specified changes take place immediately. If one of these secondary operations fails (say, due to a file system being not unmountable, due to its being in use), the modifications will still have been made to the database files.

Options for the `list` operation (`-l`, `-r` and `-e` are mutually exclusive):

`-q` Quiet. Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

`-v` Verbose. Produce a formatted listing with headers and aligned columns. This option is enabled by default.

`-l` List only local file systems.

`-r` List only remote file systems.

`-e` List only exported file systems (from `/etc/xtab`).

`-m` List only mounted file systems (from `/etc/mnttab`).

Option for the `check` operation:

`-p` *fsck-options*
Options to be passed to the `fsck` command. See `fsck`(1M) for a list of valid `fsck` command-line options.

EXAMPLE

For this example, assume that you wish to access a remote NFS file system mounted /pdd/acct on remote host, div3. The file system is to be soft-mounted locally for read-only access at directory /acct/div3. The file system would be added to `fstab` using

```
admfilesystem -o add -f div3:/pdd/acct -p "ro,soft" /acct/div3
```

It could then be mounted using

```
admfilesystem -o mount /acct/div3
```

OUTPUT

The output produced by the `list` operation with the `-e` option lists the mount directory and export options for each exported file system. For other variations of the `list` operation, the following information is reported: logical disk, mount directory, file system type, read/write permission, NFS nount type, dump cycle and fcsk pass. Information is printed in aligned columns with column headers. If the `-q` option is specified, headers are suppressed, each entry is printed on a separate line, and fields are delimited by a single space.

FILES

/etc/fstab	file system table
/etc/exports	exported file system list
/etc/mnttab	mounted file system list

DIAGNOSTICS

Warnings

- The `exports` file is inaccessible for an `add` or `modify` operation.

Errors

- *fs-source* references an unregistered host.
- *fs-source* references a non-existent logical disk.

- Attempt is made to add a remote file system whenever NFS and/or the network package has not been set up.
- Attempt is made to add a file system entry that already exists or reference a file system entry that doesn't exist.
- Attempt is made to delete or unmount either the root or the usr file system.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Super-user privilege is required for all operations except `list`.

SEE ALSO

`sysadm`(1M), `exportfs`(1M), `fsck`(1M), `mkfs`(1M), `mount`(1M), `exports`(4), `fstab`(4), `mnttab`(4), `nfs`(6P).

NAME

admfsinfo - display information about files and directories

SYNOPSIS

admfsinfo -o find [-a *days*] [-m *days*] [-n *name*] [-t *type*] [-b *bytes*] [-u *user*] [-g *group*] [-flqv] [-c *count*] [-s *how*] [*directory* ...]

admfsinfo -o check [-lqv] [*directory* ...]

admfsinfo -o diskuse [-lqv] [*directory* ...]

DESCRIPTION

admfsinfo displays information about files and file systems.

Operations

find — Display all files in *directory* and its sub-directories which match all of the specified attributes. At most *count* lines are produced. Output can be sorted either by file size, last access or modification time, or by name. If not specified, *directory* defaults to the root directory.

check — Display all files in *directory* and its sub-directories which are "suspicious" from an administrator's view point. These files should be inspected as possible administration errors or security breaches. Two problems are reported - (1) device files outside of /dev and (2) files owned by root with the setuid mode enabled. If not specified, *directory* defaults to the root directory.

diskuse — Display the total number of blocks and inodes and the number of free blocks and inodes of each mounted file system. The percentages of blocks and inodes that are in use are also reported. If *directory* is specified, statistics are reported only for the file system containing the specified directory. Otherwise, information about all file systems is reported.

Options

The following options are accepted with the find operation:

-a *days* — Minimum number of days since last access.

-m *days* — Minimum number of days since last modification.

-n *name* — Simple file name (shell wildcard characters may be used if quoted).

-t *type* — Type of file (see find(1)).

-b *bytes* — Minimum size in bytes.

-u *user* — Owner's login name or uid.

-g *group* — Owner's group name or gid.

-f — Restrict the search to the file system containing the directory.

-c *count* — Maximum number of lines of output, not including the header line (default is unlimited).

-s *how* — How to sort, one of the following:

accessed
: time of last access, oldest first

accessed:r
: time of last access, newest first

modified
: time of last modification, oldest first

`modified:r`
time of last modification, newest first
`size` file size, largest first
`size:r` file size, smallest first
`name` file name

If this option is not specified, the output will be unsorted.

All operations allow the following:

`-l` Consider only local file systems, excluding remote mounted file systems.

`-q` Quiet, do not print headers or extraneous information in reports.

`-v` Verbose, include headers and additional report information (this option is enabled by default).

OUTPUT

The `find` operation displays the following information, one line per file, for each file that meets the selection criteria:

- owner of the file
- file size in bytes
- last access day and time
- file name

The `check` operation displays the following information, one line per file, for each file with a suspected security problem:

- problem identification
- file name

The `diskuse` operation displays the following information, one line per file system:

- file system name
- number of free inodes
- total inodes
- percent of inodes in use
- number of free blocks
- total blocks
- percent of blocks in use

EXAMPLES

To show the 25 largest regular files owned by root on all local file systems, use:

```
admfsinfo -o find -l -u root -t f -s size -c 25
```

To show the 50 largest regular files in the /opt and /usr/opt file systems, last accessed more than one day ago, use:

```
admfsinfo -o find -t f -a 1 -s size -c 50 /opt /usr/opt
```

DIAGNOSTICS

Warnings

- Some (but not all) directories specified for the `find` or `check` operations do not exist of cannot be examined.

Errors

- Cannot create a temporary file.
- Directory specified for the `diskuse` operation does not exist or cannot be examined.

- All directories specified for the `find` or `check` operations do not exist or cannot be examined.

Exit Codes

0 The operation was successful.

1 No files were found to match the selection criteria; all specified directories do not exist or cannot be examined.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

REFERENCES

The `find` operation handles the *fileage* (`-m`), *filename* (`-n`), and *filesize* commands of `osysadm`(1M). *fileage* sorts by time (`-s time`) and defaults to 90 days (`-m 90`). *filesize* sorts by size (`-s size`) and defaults to printing the 10 largest (`-c 10`) regular files (`-t f`). *filename* is unsorted. The current *filesize* uses a separate utility to provide the size information for the sort.

The `check` operation handles `osysadm`'s *filescan* commands.

The `diskuse` operation handles `osysadm`'s *diskuse* command. The specification of a directory is new.

SEE ALSO

`df`(1), `du`(1), `find`(1), `sysadm`(1M).

NAME

admgroup - manage group information in the group database

SYNOPSIS

admgroup -o add [-y] [-g *gid*] [-m *member-list*] *group*

admgroup -o modify [-y] [-g *gid*] [-n *new-group*] [-m *member-list*] [-r *member-list*] [-a *member-list*] *group*

admgroup -o delete [-y] *group* ...

admgroup -o list [-qv] [-y] [*group* ...]

DESCRIPTION

admgroup manages the user and group information in the group(4) database file. The group file contains a group name that associates a numerical group id with one or more users who have access to group information.

admgroup is normally run by the system administrator on the NIS (YP) master machine if the system is running NIS, or on any host if the system is not running NIS.

Operations

add — Add a new group to the group file. If the specified *group* already exists, the operation will not be successful. Only the superuser may execute this operation.

modify — Modify currently existing information in the local or NIS group database. The command-line options will determine the changes that are made to the *group* entry in the group database. Only the superuser may execute this operation.

delete — Delete the given *group*(s) from the group database. Only the superuser may execute this operation.

list — List the group(s) in the group database which match the specified command-line options. If no command-line options are given, then only the local group database will be listed. Any user may execute this operation.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-g *gid* — The new group ID for the *group* should be *gid*. It must be a non-negative integer less than or equal to MAXUID , as defined in <sys/param.h>. It defaults to an available unique group ID above 99.

-n *new-group*
Specifies a string of printable characters that is the new group name for the *group*. It may not contain a colon (:) or newline and must be a unique group name.

-m *member-list*
Specifies the login names of users who are to be members of the *group*. This list must be a quoted space or comma-separated list of names (see example). The names must be valid user names in the passwd database.

If this option is not specified for the `add` operation, then the group will have no members.

`-a` *member-list*
Specifies the login users who are to be added to the list of members in the *group*. This list must be a quoted space or comma-separated list of names (see example). The names must be valid user names in the `passwd` database.

`-r` *member-list*
Specifies the login users who are to be removed from the list of members in the *group*. This list must be a quoted space or comma-separated list of names (see example). The names must be valid user names in the `passwd` database.

`-q` Quiet. The headers are not printed when listing group information.

`-v` Verbose. The headers are printed when listing group information. This option is enabled by default.

EXAMPLES

```
admgroup -o add -g 101 -m "sjones esmith" sysadmin
admgroup -o modify -a connor,wamo sysadmin
```

DIAGNOSTICS

Warnings

- Only one member modification option may be specified. Other options will be ignored.

Errors

- The `add`, `delete`, or `modify` operation was unable to update the group database.
- The `modify` operation could not find the specified *group*.

Exit Codes

This section lists the possible exit codes and what they mean.

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

FILES

/etc/group
Local group database file.

SEE ALSO

`groups`(1), `groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `sysadm`(1M), `group`(4).

NAME

admhost - manage hosts database

SYNOPSIS

admhost -o add [-y] [-l *alias-list*] -a *network-address host-name*

admhost -o modify [-y] [-n *new-host-name*] [-a *network-address*] [-l *alias-list*] *host-name*

admhost -o delete [-y] *host-name* ...

admhost -o list [-y] [-qv] [-s] [*host-name* ...]

DESCRIPTION

admhost manages the local or the NIS (YP) hosts(4) database. The hosts database consists of a list of host names and the Internet address and possible aliases for each.

Operations

add — Create a new hosts entry.

modify — Change a host name, network address or alias list for an existing hosts entry.

delete — Remove an entry from the hosts database.

list — List one or more entries from the hosts database. Information is listed about each of the *host-name*s given; if no *host-name*s are given or *host-name* is "all", information about all hosts is listed.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. If specified with the add, delete, or modify operations, this option is valid only if the machine on which the command is run is the NIS master. The -y option uses the default domain name derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-a *network-address*
network-address is the Internet address of the host. If this option is excluded on modify, the current Internet address for the host is preserved.

-l *alias-list*
alias-list is a comma-separated list of aliases by which the host can be referenced. If this option is excluded on modify, the current alias list for the host is preserved.

-n *new-host-name*
new-host-name is the new name of the host. Without this option, the host name is not changed.

-q — "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-s — "Sort." Sort the listing by host name.

-v — "Verbose." Produce a formatted listing with headers and aligned columns. This is the default.

admhost expects all Internet addresses to be of the form

a.b.c.d,

where *a* is a decimal number between 0 and 224, and *b*, *c*, and *d* are decimal numbers between 0 and 255.

OUTPUT

The `list` operation writes its output to stdout.

The verbose form of the `list` operation outputs the name, Internet address and alias list for each host. Information is printed in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

internet_address host_name alias1 alian2 aliasN ...

FILES

/etc/hosts Local hosts database file.

DIAGNOSTICS

Warnings

None.

Errors

- The `-y` option is specified for an `add`, `delete`, or `modify` operation and the host is not the NIS master.
- The `add` operation is requested, and *host-name* or *network-address* already exists.
- The `modify` operation is requested, and *new-host-name* or *network-address* already exists.
- The *network-address* is not a valid Internet address.
- The `-y` option is specified, and there is an error exporting NIS maps.
- The `delete` or `modify` operation is requested, and *host-name* does not exist.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

You must have write permission to the `hosts` database to use the `add`, `delete`, and `modify` operations. Usually, only the super-user has such permission.

SEE ALSO

`domainname`(1), `sysadm`(1M), `hosts`(4).

NAME

admipinterface - manage the TCP/IP network interfaces database

SYNOPSIS

admipinterface -o add -d *device* [-m *netmask*] [-b 0 | 1] [-p *link-protocol*] [-t *template*] *hostname*

admipinterface -o delete *hostname* ...

admipinterface -o modify [-n *new-hostname*] [-d *device*] [-m *netmask*] [-b 0 | 1] [-p *link-protocol*] [-t *template*] *hostname*

admipinterface -o list [-qv] [*hostname* ...]

admipinterface -o start *hostname* ...

admipinterface -o stop *hostname* ...

DESCRIPTION

admipinterface manages the TCP/IP network interfaces database. A TCP/IP network interfaces database entry consists of: a hostname (which must have an hosts(4) entry) or an Internet address, an interface name, a device name(which must have a /dev entry), a netmask, a broadcast address for broadcasting interfaces, and an IXE template for ixe(7) interfaces.

Operations

add — Add a new network interface to the network interfaces database.

delete — Remove one or more network interface entries from the network interfaces database.

modify — Change a network interface entry in the network interfaces database. The hostname, device, netmask, broadcast address, link-level protocol, and template name may be edited.

list — List one or more network interfaces in the network interfaces database. Information is listed about each of the *hostnames* given; if no *hostnames* are given or *hostname* is all, information about all network interfaces is listed.

start — Start one or more network interfaces from the network interfaces database using the ifconfig command.

stop — Stop one or more network interfaces from the network interfaces database using the ifconfig command.

Options

-d *device* — *device* is the device name which must have a /dev entry. The devices currently supported are: inen(7), hken(7), ixe(7), vitr(7), loop(6).

-m *netmask*

netmask is the hexdecimal mask that masks off the host part from the network part of the Internet address . Without this option the default netmask will be used. The default netmask does not allow subnetting.

-b 0 | 1 — *broadcast-polarity* specifies whether the host part of the broadcast address is composed of 1's or 0's. The default is 1.

-p *link-protocol*

link-protocol is the Link level protocol used to build the network protocol stack. The Link level protocols currently supported are: ether and 802.3. This option is only valid for inen and hken devices. The *link-*

protocol defaults to `ether`. If the *link-protocol* is not `ether`, then the interface name will be derived by prepending *link-protocol* onto *device*, otherwise the interface name will be the same as the device name. For example, if *device* is `hken0`, and *link-protocol* is `802.3`, then the interface name will be `802.3_hken0`. However, if *link-protocol* is `ether`, then the interface name will be `hken0`.

`-t` *template*
: *template* is the file that contains IXE information for X.25 network interfaces. This option is only valid for `ixe` interfaces.

`-n` *new-hostname*
: *new-hostname* is the new hostname that will replace *hostname* in the modify command. Without this option the *hostname* is not changed.

`-q`
: "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

`-v`
: "Verbose." Produce a formatted listing with headers and aligned columns. This option is enabled by default.

EXAMPLES

In the following examples, the system administrator performs the following actions: add the *my-host-ixe0* and *my-host-ixe1* `ixe` interfaces; add the *my-host* `hken0` interface; modify the *my-host* `hken0` interface by specifying a subnet mask.

```
admipinterface -o add -d ixe0 -m 0xffff0000 -t ixe0.file my-host-ixe0
admipinterface -o add -d ixe1 -m 0xffff0000 -t ixe1.file my-host-ixe1
admipinterface -o add -d hken0 -b 1 -p ether my-host
admipinterface -o modify -m 0xffffff00 my-host
```

FILES

/etc/tcpip.params
: File that contains the network interfaces database.

OUTPUT

The `list` operation writes its output to stdout.

The verbose form of the `list` operation outputs the entry in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

hostname interface device netmask broadcast link-protocol template

DIAGNOSTICS

Warnings

- The `delete`, `start`, or `stop` operation is requested, and *hostname* does not exist.

Errors

- The `add` operation is requested, and *hostname* already exists.
- The `modify` operation is requested, and *hostname* does not exist.
- The `modify` operation is requested, and *new-hostname* already exists.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Only the system administrator is granted access to the `add`, `delete`, `modify`, `start`, and `stop` operations. Anyone with read access to the network interfaces database file is granted access to the `list` operation.

SEE ALSO

`ifconfig`(1M), `hosts`(4), `tcpip.params`(4), `loop`(6), `inen`(7), `hken`(7), `vitr`(7), `ixe`(7) in the X.25 product.

NAME

admkernel - manipulate the system's kernel

SYNOPSIS

admkernel -o autoconfigure [-C] [-d *build-directory*] *system-name*

admkernel -o build *system-name*

admkernel -o install [-Cr] [-d *build-directory*] *system-name*

admkernel -o link [-c *client-list*] *system-name*

admkernel -o list [-qv] [-d *build-directory*] [*system-name* ...]

DESCRIPTION

admkernel builds and installs system kernels.

A system's kernel is the image which is loaded into memory when booting the system. The admkernel command manages this system's kernel and may also manage the kernels of any operating system clients of this system.

Operations

autoconfigure
: Create a system file in the *build-directory* which lists the hardware devices currently attached to the system, as well as the pseudo-devices and configuration variables of any loaded packages. The resulting system file, system.*system-name*, can be used to build a custom kernel for the system.

build
: Build a new system kernel from an existing system file in the *build-directory*. This includes running the config(1M) program on a system file and compiling the resulting file (with cc(1) and ld(1)) with the kernel libraries into a bootable kernel image, called dgux.*system-name*. Note that kernels can only be built for the PRIMARY release area.

install
: Copy a kernel from the *build-directory* into some other directory where it can be booted as the system's kernel. If the operation is successful, the kernel is deleted from the build directory.

link
: Link a kernel to the default kernel (/dgux) for this host or for clients of this host.

list
: Display information about *system-name*s. If there is no *system-name* or the *system-name* is all, the list of valid *system-names* is displayed.

Options

-C
: Perform the operation for an operating system client. For the autoconfigure operation, this option indicates that the kernel will be used by an operating system client. For the install operation, this option indicates that the kernel will be installed for an operating system client of this host.

-c *client-list*
: Link a kernel as the default kernel for each client in *client-list*. *client-list* is either a comma-separated list of clients, or may be "all" to indicate that the kernel should be linked for all clients of this host.

-d *build-directory*
: The directory in which system files and new kernels reside. The default is /usr/src/uts/aviion/Build.

-r
: Remove the old kernel (dgux.*system-name*), if it exists. Without this option, the old kernel is saved to dgux.*system-name*.old before the new

kernel is installed.

`-q` Quiet. Omit header lines from the output of the `list` operation.

`-v` Verbose. Include header lines in the output of the `list` operation. This option is enabled by default.

EXAMPLES

To generate a system file based on the system's current configuration, use

```
admkernel -o autoconfigure aviion
```

To build, install, and link a kernel based on the "aviion" system file, use

```
admkernel -o build aviion
admkernel -o install aviion
admkernel -o link aviion
```

which builds `dgux.aviion` and links it to `/dgux`.

To build, install, and link a kernel for two diskless clients of this host, first create `/usr/src/uts/aviion/Build/system.diskless` with the appropriate parameters, then use these command lines

```
admkernel -o build diskless
admkernel -o install -C diskless
admkernel -o link -c myclient,yourclient diskless
```

ENVIRONMENT VARIABLES

Several environment variables are used to control which tools are used for building new kernels. Some of the more useful variables are listed below.

CC Name of the compiler, with its options, to use when compiling the configuration file into an object file. The default is "/usr/bin/gcc -mno-underscores".

LD Name of the linker, with its options, to use when linking the kernel libraries to form the kernel image. The default is "/usr/bin/ld -x -F 0555".

LIB_DIR Directory where kernel libraries are found. Define this variable to build with a non-default set of libraries. The default is "/usr/src/uts/aviion/lb".

You should exercise great care when defining these variables in your environment, because some combinations of settings may cause the resulting kernel image to be unbootable.

FILES

/usr/src/uts/aviion/Build
Build directory.

/usr/src/uts/aviion/cf
Directory containing prototype system files.

/dgux.*system-name*
Name of the installed kernel for this host.

/srv/release/PRIMARY/_Kernels/dgux.*system-name*
Name of the installed kernel for operating system clients of this host.

OUTPUT

The `list` operation displays the system name and date of last modification for each of the *system-name*s.

DIAGNOSTICS

Warnings

- The file `system.`*system-name* does not exist. It will be created from the prototype files in `/usr/src/uts/aviion/cf`.

Errors

- There is no C compiler on the system.
- One of the necessary directories under `/usr/src/uts/aviion` is missing or incomplete.
- The kernel failed to configure (with `config`(1M)) or failed to link (with `ld`(1)).
- One of the clients in the *client-list* does not exist.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`cc`(1), `config`(1M), `ld`(1), `probedev`(1M), `sysadm`(1M), `system`(4).

NOTES

The `autoconfigure` operation uses `probedev`(1M) to determine which hardware devices to include in the system file.

This command includes the functions of the `osysadm`(1M) *newdgux* command.

The `autoconfigure` and `build` operations require write permission in the *build-directory1. The* `install` and `link` operations require write permission in the target root directory. Normally, these four operations require superuser privilege.

NAME

admlock - manage simple process synchronization

SYNOPSIS

admlock -o create [-qv] [-w *time-limit*] *lock-name*

admlock -o remove [-qv] *lock-name*

admlock -o check [-qv] [*lock-name*...]

admlock -o list [-qv] [*lock-name*...]

where:

time-limit The number of seconds to wait for a lock

lock-name A string that conforms to simple filename rules. If no *lock-name* is specified for the `check` and `list` operations, all known locks will be examined.

DESCRIPTION

`Admlock` manages a simple process synchronization mechanism where the existance of a named file acts as an advisory lock. Cooperating processes may use the lock to control access to critical resources.

`Admlock` is intended for use in shell scripts, providing them an atomic test and set mechanism. Each shell script or set of shell scripts that will cooperate in managing a resource should select a *lock-name*. By testing and setting the lock before manipulating a critical resource, and releasing the lock afterwards, concurrent processes can maintain the resource's integrity.

Operations

`create` Test and set a lock using the specified lock name. A file by that name will be created in `/etc/sysadm/locks`.

`remove` Release a lock by removing the corresponding lock file.

`check` Examine a lock to make sure that the locking process is still active. If the locking process is not active, the lock will be released.

`list` List information about one or more locks.

Options

`-w` Specify a time limit to wait for a lock. This option's value should be based on the expected length of time a cooperating process might hold the lock, and is provided to guard against accidental persistance of a lock after the locking process goes away. A zero value implies that the caller should not wait at all if the lock is already set.

`-q` Quiet, do not print headers for the `list` operation or error messages from other operations. This is the default for all operations except `list`.

`-v` Verbose, include header lines in the `list` output and error messages from other operations. This is the default for the `list` operation.

EXAMPLE

To test and set a lock:

```
if admlock -o create -w 30 mylockname
then
    :
else
    echo "could not get lock in 30 seconds"
    exit 1
fi
```

To release a lock when done:

```
admlock -o remove mylockname
```

DIAGNOSTICS

Exit Codes

0 The lock was created (`create`) or released (`remove`) successfully. The `check` operation returns this value if the specified lock file does not exist, or an existing lock file which did not have an active process associated with it was successfully removed, or the locking process is on a remote system and cannot be checked.

1 The lock could not be created within the time limit restrictions (`create`), or an existing lock has an active locking process (`check`).

2 A previous lock does not exist but a lock file could not be created (`create`), a lock file owned by this process could not be released (`remove`), or the specified lock file does not have an active process associated with it and the lock file was not or could not be removed (`check`).

3 There was an error in the command line.

ENVIRONMENT

The following environment variables are used:

ROOT An alternate root pathname, if not "/". This is provided for use by a server system.

FILES

`/etc/sysadm/locks`
Directory containing lock files

SEE ALSO

`fcntl`(2), `semop`(2).

NOTES

`Admlock` works by attempting to create the named file. If the creation fails because the file already exists, the command waits a short time and tries again. As failure persists, the command waits longer and longer time periods until the time limit expires (it waits at most 30 seconds between tries). At each failure, a check is made to ensure that the locking process continues to exist, and the old lock is released if the locking process is no longer active. When the file creation succeeds, the following information is written to the lock file using the indicated format:

`"%5d %-12s %15d\n"`, *parent-pid, host-name, time-since-epoch*

This information is used by the `check` operation and by the `create` operation to ensure that the calling process does not already have the file locked.

Any signal received while waiting to create a lock file will cause the command to exit with an exit code of 1.

File locking using `fcntl`(2) is not involved in this procedure.

BUGS

Deadlock detection is not attempted, except in the case where a process attempts to lock a lock it already has.

NAME

admnetwork - manage network database

SYNOPSIS

admnetwork -o add [-y] [-l *alias-list*] -a *network-address network-name*

admnetwork -o modify [-y] [-n *new-network-name*] [-a *network-address*] [-l *alias-list*] *network-name*

admnetwork -o delete [-y] *network-name* ...

admnetwork -o list [-y] [-qv] [-s] [*network-name* ...]

DESCRIPTION

admnetwork manages the local or the NIS (YP) network database. The network(4) database consists of a list of network names and the Internet address for each.

Operations

add — Add a new network entry.

modify — Change an existing entry to contain a new network name or a new network address, or both.

delete — Remove an entry from the network database.

list — List one or more entries from the network database. Information is listed about each of the *network-name*s given; if *network-name* is "all", information about all networks is listed.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. If specified with the add, delete, or modify operations, this option is valid only if the machine on which the command is run is the NIS master. The -y option uses the default domain name derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-a *network-address*
network-address is the Internet address of the network. If this option is excluded on modify, the current Internet address for the network is preserved.

-l *alias-list*
alias-list is a comma-separated list of aliases by which the network can be referenced. If this option is excluded on modify, the current alias list for the network is preserved.

-q — "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-s — "Sort." Sort the listing by network name.

-v — "Verbose." Produce a formatted listing with headers and aligned columns. This is the default.

admnetwork expects all network addresses to be of the form

a.b.c.d,

where *a* is a decimal number between 0 and 224, and *b*, *c*, and *d* are decimal

numbers between 0 and 255. The address must have all four octets.

OUTPUT

The `list` operation writes its output to stdout.

The verbose form of the `list` operation outputs the name and Internet address for each network. Information is printed in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

network_name internet_address

FILES

/etc/networks

Local networks database file.

DIAGNOSTICS

Warnings

None.

Errors

- The `-y` option is specified for an `add`, `delete`, or `modify` operation and the host is not the NIS master.
- The `add` operation is requested, and *network-name* or *network-address* already exists.
- The `modify` operation is requested, and *new-network-name* or *network-address* already exists.
- The `delete` or `modify` operation is requested, and *network-name* does not exist.
- The *network-address* is not a valid network address.
- The `-y` option is given, and there is an error exporting NIS maps.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line. does not exist.

NOTES

You must have write permission to the `networks` database to use the `add`, `delete`, and `modify` operations. Usually, only the super-user has such permission.

SEE ALSO

`domainname`(1), `sysadm`(1M), `networks`(4).

NAME

admnls - manipulate national language variables

SYNOPSIS

admnls -o set *parameter=value ...*

admnls -o get [-qv] [*parameter ...*]

DESCRIPTION

The admnls command manages national language support (NLS) environment variables.

The following environment variables are managed:

LANG The user's chosen locale. The default is C.

NLSPATH The path on which to look for message catalogs. The default is /usr/lib/nls/msg/%L/%N:/etc/nls/msg/%L/%N.

The NLSPATH variable is used by catopen(1) to locate X/Open message catalogs. NLSPATH can consist of a series of pathnames, separated by colons. This is useful for programs which install message catalogs somewhere other than the default location For example, if the hypothetical ipslng product installs message catalogs under /usr/opt/ipslng/*locale*, NLSPATH could be set to /usr/lib/nls/msg/%L/%N:/usr/opt/ipslng/%L/%N. In NLSPATH, the sequence "%L" is replaced with the value of the environment variable LANG.

The LANG variable is also used by setlocale(3) to determine which subdirectory of /usr/lib/locale to use for other locale-dependent information.

Operations

The following operations are provided:

set Assign new values to one or more parameters.

get Retrieve the values for one or more parameters. If *parameter* is not given, all parameters are displayed. Otherwise, *parameter* must be one of LANG or NLSPATH.

Options

The following options are provided:

-q Use quiet mode. For the get operation, there are no headers and only the values of the requested parameters are displayed.

-v Use verbose mode. For the get operation, headers are displayed. This option is enabled by default.

EXAMPLES

For example, to set the LANG variable to C and the NLSPATH variable to /usr/lib/nls/msg/%L/%N, use a command line like

```
admnls -o set LANG=C NLSPATH="/usr/lib/nls/msg/%L/%N"
```

FILES

/etc/TIMEZONE
Contains the values of the variables for sh(1) users.

/etc/TIMEZONE.csh
Contains the values of the variables for csh(1) users.

DIAGNOSTICS

Warnings

- Either database file (/etc/TIMEZONE or /etc/TIMEZONE.csh) is missing. The file will be created from the prototype.
- Either of `LANG` or `NLSPATH` is missing from the database file. It will be added.

Errors

- There is an error modifying one of the database files.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Any logins and processes running when environment variables are changed, and all their child processes, will continue to see the old values of the variables. In order to ensure that all processes run with the new values for `LANG` and `NLSPATH`, you must reboot the system.

You must have permission to write to the /etc/TIMEZONE file in order to use the `set` operation. Normally, only super-user has such permission.

REFERENCES

X/Open Portability Guide, Volume 3.

SEE ALSO

`catgets`(1), `csh`(1), `gettxt`(1), `sh`(1), `catgets`(3), `catopen`(3), `gettxt`(3), `localeconv`(3), `setlocale`(3), `timezone`(4), `environ`(5).

NAME

admpackage - manage DG/UX-style software packages

SYNOPSIS

admpackage -o load [-qv] [-r *release-area*] [-f *release-medium*] [*package ...*]

admpackage -o setup [-r *release-area*] [-c *client-list*] [*package ...*]

admpackage -o install [-qv] [-r *release-area*] [-f *release-medium*] [-c *client-list*] [*package ...*]

admpackage -o list [-qvR] -f *release-medium*

admpackage -o list [-qv] [-r *release-area*] [*package ...*]

admpackage -o list [-qv] -c *client-list* [-r *release-area*]

DESCRIPTION

admpackage is used for loading and performing necessary setup of DG/UX packages. All software products that come from Data General for the AViiON series computers are considered to be DG/UX packages. The software release format is a superset of that used for SunOS 4.0. Only the super-user may run this command, except for the list operation, which may be run by any user.

Operations

load — Load the specified package(s) from the *release-medium* into a release area on the disk. If no packages are specified on the command line, then all packages on the *release-medium* will be loaded.

setup — Execute the package's setup script(s), stored in the specified release area, to prepare the package for use. If no packages are specified, then all packages that have not previously been set up will be set up.

install — This operation combines the load and setup operations into a single step. Note that all packages will first be loaded, then all packages will be set up. If a package does not have setup scripts, then no action will be taken for the setup step.

list — The list operation has three forms. In the first form, distinguished by the -f option, the operation will list to the standard output the packages which exist on the specified *release-medium*. For the second form, if no options are specified, or only a release area is specified, then the operation will list to the standard output the packages which have been installed in a given release area. A package is considered to be installed when it has been loaded and set up. In the final form, distinguished by the -c option, the operation will list packages that are loaded but require set up for the specified list of operating system clients. You may use MY_HOST or the output from hostname(1C) for the client list if you want to list packages on your host that require set up. For all forms of this operation, the -q option is automatically enabled unless the -v option is specified.

Options

-f *release-medium*
Specify the medium from which the packages will be loaded. The default *release-medium* is obtained through admdefault(1M). The medium may be the name of a tape device or a directory.

-r *release-area*
Specify the release area into which the package is to be (or was) installed.

The default release area is PRIMARY.

-c *client-list*
Specify the list of operating system clients for which the package is to be set up. This list must be a comma-separated list of clients that are served by your host. The default client list is MY_HOST.

-q Quiet. Print minimal information about the files on the system or the *release-medium*.

-v Verbose. Print detailed information about the files on the system or *release-medium*.

-R Print release information for the *release-medium* itself in addition to information about the files on the *release-medium*.

Loading from Disk

The *release-medium* value may be a disk directory name instead of a tape device name. When this is the case, admpackage expects to find a table of contents file named 1.xdrtoc in that directory. This file is the same file that would appear as the second file on a distribution tape, and contains names of other images. These other images must also appear in the disk directory with filenames composed of the "tape" file number, a period, and the image name from the table of contents file. For example, if the table of contents file says that file 4 has the image "tcpip__r.base", then the directory should contain a file named 4.tcpip__r.base.

DIAGNOSTICS

Warnings

- The load operation is requested and the tape contains a file load destination that is not /, or a part of /usr or /opt.
- The load operation is requested and admpackage was unable to update the root prototype directory.
- The load operation is requested and the database file for the client being updated is missing.
- The setup operation is requested and the setup script failed. The setup script should provide more specific information regarding the failure.

Errors

- admpackage is unable to read or manipulate the *release-medium*.
- The *release-medium* does not contain a table of contents.
- admpackage is unable to create necessary temporary files. The root file system is probably out of space.
- The table of contents found on the tape is in an invalid format.
- The specified *release-area* does not exist or is invalid. The user must create a release area before loading packages into that release area (see admrelease(1M)).

Exit Codes

This section lists the possible exit codes and what they mean.

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

For all of the operations that optionally take a *package* argument, the key word `all` may be used to perform the operation on all of the available packages.

SEE ALSO

`admclient`(1M), `admdefault`(1M), `admrelease`(1M), `admtape`(1M), `sysadm`(1M).

NAME

admportmonitor - manage port monitors

SYNOPSIS

admportmonitor -o add -t *type* [-c *command*] [-e | -d] [-r *rst-count*] [-s | -p] [-x *version*] [-y *comment*] [-z *script*] *pmtag*

admportmonitor -o delete *pmtag* ...

admportmonitor -o disable *pmtag* ...

admportmonitor -o enable *pmtag* ...

admportmonitor -o modify [-e | -d] [-r *rst-count*] [-s | -p] [-z *script*] *pmtag*

admportmonitor -o start *pmtag* ...

admportmonitor -o stop *pmtag* ...

admportmonitor -o list [-qv] [-t *type* | *pmtag* ...]

DESCRIPTION

Beginning with DG/UX Release 5.4, the Service Access Facility generalizes the procedures for service access so that login access on a local system and network access to local services are managed in essentially the same way.

admportmonitor manages port monitors under control of sac(1M) (the Service Access Controller for the Service Access Facility). The Service Access Controller is the overseer of the server machine.

sac is the Service Access Facility's controlling process and is started by init(1M). Its function is to maintain the port monitors on the system in the state specified by the system administrator. These states include: STARTING, ENABLED, DISABLED, STOPPING, NOTRUNNING, and FAILED. (A port monitor enters the FAILED state if sac cannot start it after a specified number of tries.) A port monitor, from sac's point of view, is simply a process with which it communicates in a well-defined way.

sac is responsible for starting port monitors, attempting to restart them whenever they fail, and passing state-change requests to them. Each port monitor process managed by sac is identified by a unique, alphanumeric name referred to as the port monitor tag (*pmtag*).

DG/UX provides two types of port monitors, ttymon(1M) and listen(1M), which are described briefly below. Systems may also contain other types of port monitors, including port monitors written expressly for a user's application.

A ttymon port monitor manages local access to the login(1) service--although it is not limited to the login service. It performs the functions which were formerly handled by getty(1M) in releases prior to DG/UX 5.4. Unlike getty, however, a single ttymon port monitor can support multiple ports. A ttymon port monitor is responsible for monitoring these ports and for invoking the service associated with a given port (e.g. login) when it it receives a connection request on that port. (The login service is the most commonly invoked service for a port; however, a port may be configured to invoke alternative services.)

A listen port monitor manages a TLI-based, connection-oriented transport network. It is responsible for receiving incoming connection requests, accepting them, and invoking the services that have been requested.

Operations

`add` Add a new port monitor to the list of managed port monitors in the `sac` administrative file (`/etc/saf/_sactab`). This operation may also start and enable the new port monitor based on the options specified.

`delete` Remove one or more port monitors from the set of port monitors administered by `sac`. If a port monitor that is to be removed is currently running, it will be stopped.

`disable` Disable one or more running port monitors. A disabled port monitor continues to run but denies any service requests it receives from the port(s) it is monitoring.

`enable` Enable one or more disabled port monitors.

`modify` Change the attributes of a port monitor.

`start` Start one or more stopped port monitors.

`stop` Stop one or more currently running port monitors. Stopping a port monitor terminates its process.

`list` List information about one or more port monitors. If `-t` *type* is given, information is listed about all port monitors of that type. If a list of port monitors is given, information is listed about each of the *pmtag*s listed; if *pmtag* is `all`, information is listed about all port monitors in the `sac` administrative file.

Options

`-c` *command*

The command string which is to be executed to start the port monitor. If `-c` is not specified and a `ttymon` or `listen` port monitor is being added, *command* defaults to `/usr/lib/saf/ttymon` or `/usr/lib/saf/listen`, respectively. The `-c` option is required when adding port monitors of all other types.

`-e` and `-d`

These options are used to specify what the initial state of the port monitor should be when it is started. If `-e` is specified, the port monitor will start in the ENABLED state. If `-d` is specified, the port monitor will start in the DISABLED state. If neither of these options is specified for the `add` operation, `-e` is implied.

`-q` "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by colons).

`-r` *rst-count*

Restart count. Normally, `sac` (the Service Access Controller) attempts to restart port monitors that terminate unexpectedly. The restart count indicates the number of times `sac` should attempt to restart the port monitor before giving up and setting its state to FAILED. The default *rst-count* is 0.

`-s` and `-p`

These options are used to specify whether the port monitor should be started. If `-s` is specified, the port monitor will be started immediately when it is added via the `add` operation and whenever `sac` is initially invoked by `init` at system startup time. If `-p` is specified, the port monitor will not be automatically started by `sac`; it must, instead, be explicitly started by the system administrator. If neither of these options is specified

for the `add` operation, `-s` is implied.

`-t` *type* Specifies the port monitor type (e.g. ttymon, listen).

`-v` "Verbose." Produce a formatted listing with headers and aligned columns. This is the default output format.

`-x` *version* Specifies the version number of the port monitor. This version number may be given as

-x '*pmspec* -V'

where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm`(1M) for `ttymon` and `nlsadmin`(1M) for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a -V option.

If a `ttymon` or `listen` port monitor is being added and `-x` is not specified, *version* defaults to the version number returned by `ttyadm -V` or `nlsadmin -V`, respectively. The `-x` option is required when adding port monitors of all other types.

`-y` *comment*
Include *comment* in the `_sactab` entry for port monitor *pmtag*.

`-z` *script* Specifies the name of a configuration script to be run when the port monitor is started. This configuration script can be used to set the environment for the port monitor. See `doconfig`(3N) for more information.

Output

The `list` operation reports the following port monitor information to stdout: name, type, flags, restart count, state, invoking command and comments. With the "verbose" (-v) format, information is printed in aligned columns with column headers. With the "quiet" format (-q), headers are suppressed and each port monitor entry is printed on a separate line. Fields within each entry are delimited by a colon and are in the following order:

Field	Description
1	port monitor tag
2	port monitor type
3	flags (d = do not enable, x = do not start)
4	restart count
5	state (STARTING, ENABLED, DISABLED, STOPPING, NOTRUNNING, FAILED)
6	command used to invoke the port monitor optionally followed by "#" and a comment

FILES

/etc/saf/_sactab
`sac` administrative file

/etc/saf/*pmtag*/_config
Per-port monitor configuration file.

DIAGNOSTICS

Errors

`admportmonitor` uses `sacadm`(1M) to perform the requested operation for the port monitor(s). It reports any error conditions returned by `sacadm`.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`login`(1), `admportservice`(1M), `listen`(1M), `nlsadmin`(1M), `sac`(1M), `sacadm`(1M), `ttyadm`(1M), `ttymon`(1M), `doconfig`(3N), *Managing the DG/UX System*.

NOTES

All operations except for `list` require superuser access.

NAME

admportservice - manage port monitor services

SYNOPSIS

admportservice -o add -p *pmtag* [-i *id*] [-m *pm-specific*] [-e | -d] [-w | -u] [-x *version*] [-y *comment*] [-z *script*] *svctag*

admportservice -o delete -p *pmtag svctag* ...

admportservice -o disable -p *pmtag svctag* ...

admportservice -o enable -p *pmtag svctag* ...

admportservice -o modify -p *pmtag* [-i *id*] [-m *pm-specific*] [-n *new-svctag*] [-y *comment*] [-z *script*] *svctag*

admportservice -o list [-qv] [-t *type* | -p *pmtag*] [*svctag* ...]

DESCRIPTION

Admportservice manages individual services for port monitors under control of the Service Access Facility. Under the Service Access Facility, each port monitor is responsible for monitoring one or more ports for connection requests. When a connection request is received on a port, the port monitor invokes the service associated with that port. A service for a port monitor may be in one of two states, ENABLED or DISABLED. When a service is disabled, all connection requests on the port with which it is associated are denied. Each port service managed by a port monitor is identified by a unique alphanumeric tag referred to as the service tag (*svctag*).

DG/UX provides two types of port monitors, ttymon(1M) and listen(1M) which are described briefly below. Systems may also contain other types of port monitors, including port monitors written expressly for a user's application.

A ttymon port monitor manages local access to the login(1M) service--although it is not limited to the login service. It performs the functions which were formerly handled by getty(1M) in releases prior to DG/UX 5.4. Unlike getty, however, a single ttymon port monitor can support multiple ports. A ttymon port monitor is responsible for monitoring these ports and for invoking the service associated with a given port (e.g. login) when it it receives a connection request on that port. (The login service is the most commonly invoked service for a port; however, a port may be configured to invoke alternative services.)

A listen port monitor manages a TLI-based, connection-oriented transport network. It is responsible for receiving incoming connection requests, accepting them, and invoking the services that have been requested.

Operations

add — Add a new service to the list of services in the port monitor's administrative file, /etc/saf/*pmtag*/_pmtab. If the -e option is specified, this operation will also enable the new service.

delete — Remove a service from the list of services in the port monitor's administrative file. If the service to be removed is currently enabled, it will be disabled by this operation.

disable — Disable a service for a port monitor.

enable — Enable a service for a port monitor.

modify — Change attributes or configuration of a service.

list — List information about one or more port monitor services. If -t *type* is given, information is listed about all services for port monitors of that

type. If a port monitor *pmtag* is given, information is listed about all of the services for that port monitor. If both a *pmtag* and an *svctag* are specified, information is listed about that particular service.

Options

`-e` and `-d`
These options are used to specify what the initial state of the service should be when it is added or when the port monitor is started. If `-e` is specified, the service will be in the ENABLED state. If `-d` is specified, the service will be in the DISABLED state. If neither of these options is specified for the `add` operation, `-e` is implied.

`-i` *id* *id* is the identity that is to be assigned to service *svctag* when it is started. *id* must be a valid user name from the `passwd`(4) database. If -i is not specified for the `add` operation, *id* defaults to "root."

`-m` *pm-specific*
Port monitor-specific information to be placed in administrative entry for the service in the port monitor's administrative file.

In general, each type of port monitor provides a command that takes port monitor-specific data as arguments and outputs these data in a form suitable for storage in the administrative file. Normally, the value for *pm-specific* will provided via a backquoted string containing a call to one of these port monitor-specific commands. The port monitor-specific commands for `ttymon` and `listen` port monitors `ttyadm`(1M) and `nlsadmin`(1M) respectively.

`-n` *new-svctag*
New name for the service.

`-p` *pmtag* Port monitor which manages the service.

`-q` "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by colons).

`-w` and `-u`
These options are used to specify whether a utmp entry should be created for the service. If `-w` is specified, no utmp entry will be associated with the service. If `-u` is specified, a utmp entry will be created for the service. If neither of these options is specified for the `add` operation, -w is implied.

`-x` *version* Specifies the version number of the port monitor administrative file. This version number may be given as

-x '*pmspec* -V'

where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm` for `ttymon` and `nlsadmin` for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a -V option.

`-t` *type* Specifies the port monitor type (e.g. ttymon, listen).

`-v` "Verbose." Produce a formatted listing with headers and aligned columns. This is the default output format.

-y *comment*
Include *comment* in the administrative entry for service *svctag*.

-z *script* Specifies the name of a configuration script to be run when the service is invoked. This configuration script can be used to set the environment for performing the service. See doconfig(3N) for more information.

Output

The list operation reports the following port monitor service information to stdout: port monitor name (*pmtag*), port monitor type, service name (*svctag*), flags, id, port monitor-specific information. With the "verbose" (-v) format, information is printed in aligned columns with column headers. With the "quiet" format (-q), headers are suppressed and each port monitor entry is printed on a separate line. Fields within each entry are delimited by a colon and are in the following order:

Field	Description
1	port monitor tag
2	port monitor type
3	service tag
4	flags (x = do not enable, u = create utmp)
5	user id under which service is to be run
6-8	reserved
9	port monitor-specific information optionally followed by "#" and a comment

FILES

/etc/saf/*pmtag*/_pmtab	Port monitor administrative file.
/etc/saf/*pmtag*/*svctag*	Per-service configuration file.

DIAGNOSTICS

Errors

admportservice uses pmadm(1M) to perform the requested operation for the port service(s). It reports any error conditions returned by pmadm.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

login(1), admportmonitor(1M), listen(1M), nlsadmin(1M), passwd(4), sac(1M), sacadm(1M), ttyadm(1M), ttymon(1M), doconfig(3N), utmp(4), *Managing the DG/UX System*.

NOTES

All operations except for list require superuser access.

NAME

admprocess - manage processes

SYNOPSIS

admprocess -o modify { -p *proclist* | -t *termlist* | -u *uidlist* } -r *priority*

admprocess -o signal { -p *proclist* | -t *termlist* | -u *uidlist* } -s *signo*

admprocess -o delete { -p *proclist* | -t *termlist* | -u *uidlist* }

admprocess -o list [-p *proclist* | -t *termlist* | -u *uidlist*] [-lqv]

DESCRIPTION

The admprocess command is used to display information about all or selected active processes, delete (terminate) processes, send a signal to processes, or change the priority of processes.

Operations

modify Change the priority of selected processes. Only the super-user can improve the priority of a process.

signal Send a signal to selected processes. Any signal can be sent, as specified by the -s option.

delete Terminate selected processes. The process is sent a SIGTERM signal, and if that fails is sent a SIGKILL signal.

list Display information about all or selected active processes. The information that is displayed is that of the ps(1) command and is described in its manual entry.

Options

-p *proclist*
A process ID, or comma-separated list of process IDs, that are to be selected.

-t *termlist*
A tty number, or comma-separated list of tty numbers, that are to be selected.

-u *uidlist* A user name or login ID, or comma-separated list of user names and login IDs, that are to be selected.

-r *priority*
The *priority* value ranges from 0 (top priority) to 39 (least priority), with 20 being the usual default. This value is 20 more that the *nice value* to be assigned to the process, from which the execution priority is computed. The value is reported using the list operation and the -l option in the column headed NI.

-s *signo* The number of the signal to be sent to the process. Signal numbers are defined in /usr/include/sys/signal.h.

-l Use the long listing, as described for the -l option to ps(1). The default is to produce a report using the -f option of ps.

-q Quiet. Header lines are omitted from list output.

-v Verbose. Header lines are included in list output. This option is enabled by default.

One of the -p, -t, or -u options is required for delete, signal, and modify. The list operation displays information about all processes if none of those process

selection options are specified.

EXAMPLES

To show all processes on /dev/tty05, use:

```
admprocess -o list -t05
```

To terminate process IDs 512 and 514, use:

```
admprocess -o delete -p 512,514
```

OUTPUT

The output of the `list` operation is that of the `ps`(1) command, and is explained in that manual entry. Without the `-v` option, the output is that of `ps -f`. With the `-v` option, the output is that of `ps -l`.

DIAGNOSTICS

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

4 No processes matched the selection criteria.

SEE ALSO

`ps`(1), `kill`(1), `renice`(1M), `sysadm`(1M), `who`(1), `kill`(2), `signal`(2).

NAME

admrelease - manage software release areas

SYNOPSIS

admrelease -o create [-g *share-directory*] [-r *root-directory*] [-s *swap-directory*] [-u *usr-directory*] *release-area*

admrelease -o delete *release-area* ...

admrelease -o list [-qv] [-f *format*] [*release-area* ...]

DESCRIPTION

The admrelease command is used to manipulate software release areas. A release is a collection of software packages intended for a specific architecture and operating system. Multiple versions of the operating system software or application packages can be installed on a single machine by using several independent release areas. A release area is a directory tree which contains the host-independent portion of a release (the /usr part) as well as a prototype of the host-specific part (the root). When client machines are associated with a release, the host-specific prototype is copied to a private portion of the release area reserved for use by that host, and the host-independent portion is shared with other clients.

The PRIMARY release area is created automatically and cannot be deleted. This release area usually holds the main operating system on servers and stand-alone machines. Other releases are secondary. They have identifying names and other attributes assigned by the system administrator during the create operation

After a release area is created, software can be loaded into it using admpackage(1M). Client machines are associated with a release area using admclient(1M).

Operations

create Construct a new release area. This construction involves making several directory trees. A release area name must be specified, as well as path-names to be used for /usr software, client roots, client swap files, and shared software.

delete Erase files and remove the directory trees associated with a release area that was created by the create operation. The PRIMARY release area cannot be deleted. Release areas that are still in use cannot be deleted. (Use the admclient's delete operation to disassociate client machines from a release area.)

list Provide information about release areas. If release area names are given, all the information about a release is displayed, including its directory structure and installed packages. If all is specified instead of a release area name, a short list of all release areas and their /usr directories is displayed, or just the release area names are displayed if the -q option is used too.

Options

Options recognized by the create operation are:

-g *share-directory*
The name of the directory for shared software. If not specified, the default is /srv/share.

-r *root-directory*
The name of the directory for the clients' root directory parent. If not specified, the default is /srv/release/*release-area*/root. The root

(host-specific) directory for a client will be named for the client and appear under this directory.

`-s` *swap-directory*
The name of the directory for clients' swap space. If not specified, the default is `/srv/swap`. A file for each client, named for the client, will be created under this directory to serve as the client's swap area.

`-u` *usr-directory*
The name of the directory for the `/usr` file system. If not specified, the default is `/srv/release/`*release-area*`/usr`. The prototype of the host specific portion of the release will eventually get loaded into the `root.proto` directory under this directory.

Options recognized by the `list` operation are:

`-f` *format* Select the list format. Possible values are

`long` All information about release areas (default except when listing all release areas).

`short` Release area name and `/usr` directory root (default when listing all release areas).

`names` List only the names of release areas.

`clients` List only the clients of the indicated release areas.

`packages`
List only the packages that have been installed in the indicated release areas.

`-q` Quiet. Do not include header lines.

`-v` Verbose. When used with `all`, a header line is displayed.

EXAMPLES

To create a new release area named *version2* with the default directory structure, use

```
admrelease -o create version2
```

FILES

/srv/release/*release-area*	Release area root
/srv/admin/releases	Release database

OUTPUT

The `list` operation writes to stdout. The `long` format shows the directory structure, installed packages, and attached clients. The `short` format shows only the release area name and the name of the `/usr` file system for each release. Other formats show only the information requested.

DIAGNOSTICS

Warnings

Attempts to delete a release area that does not exist are indicated and the operation on that release area is skipped.

Errors

It is not possible to delete the `PRIMARY` release area or a release area which still has clients associated with it.

It is an error to add a release area that already exists.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

The `/srv` directory structure must be established appropriately before `admrelease` or `admpackage` can function. Both commands will check the directory structure and establish the `/srv` directory tree if necessary.

Only the super-user may use the `create` and `delete` operations.

SEE ALSO

`admclient`(1M), `admpackage`(1M), `sysadm`(1M).
Installing the DG/UX System, *Managing the DG/UX System*.

NAME

admresolve - manage DNS resolver's domain name and nameservers database

SYNOPSIS

admresolve -o set *domain-name*

admresolve -o get [-qv]

admresolve -o add *nameserver*

admresolve -o delete *nameserver* ...

admresolve -o modify -n *new-nameserver nameserver*

admresolve -o list [-qv] *nameserver* ...

DESCRIPTION

admresolve manages parameters for the resolver(3) including the domain name and the nameservers database. The nameservers database consists of up to a maximum of three nameservers. Each nameserver entry consists of the hostname or Internet address of the nameserver.

Operations

set — Set the DNS domain name for the resolver.

get — Get the DNS domain name for the resolver.

add — Add a new nameserver to the resolver's DNS nameservers database.

delete — Delete one or more nameservers from the resolver's DNS nameservers database.

modify — Modify a nameserver from the resolver's DNS nameservers database.

list — List one or more nameservers from the resolver's DNS nameservers database.

Options

-n *new-nameserver*
: *new-nameserver* is the new nameserver that replaces *nameserver* in the modify operation.

-q
: "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-v
: "Verbose." Produce a formatted listing with headers and aligned columns. This is enabled by default.

admresolve expects all Internet addresses of the form

a.b.c.d,

where *a* is a decimal number between 0 and 224, and *b*, *c*, and *d* are decimal numbers between 0 and 255.

FILES

/etc/resolv.conf
: DNS resolver file.

OUTPUT

The list operation writes its output to stdout. The verbose form of the list operation outputs information in aligned columns with column headers. If -q option is specified with the list operation, headers are suppressed and each nameserver is printed on a separate line.

DIAGNOSTICS

Warnings

- The `delete` operation is requested, and *nameserver* does not exist.
- The `get` operation is requested, and *doamin-name* is not set.

Errors

- The `add` operation is requested, and the *nameserver* already exists.
- The `add` operation is requested, and the maximum number of nameservers already exist.
- The `modify` operation is requested, and the *nameserver* does not exist.
- The `modify` operation is requested, and the *new-nameserver* already exists.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line. does not exist.

NOTES

Access to the `set`, `add`, `delete`, and `modify` operations is based on write access on the /etc/resolv.conf file. Generally only the system administrator has write access. Access to the `get` and `list` operations is based on the read access on the `/etc/resolv.conf` file. Generally, all users have read access.

SEE ALSO

`resolver`(3), `resolv.conf`(4).

NAME

admroute - manage routing databases

SYNOPSIS

admroute -o add [-p] [-t net | host] -g *gateway* [-m *metric*] *destination*

admroute -o modify [-p] [-t net | host] [-d *new-destination*] [-G *new-gateway*] [-m *metric*] -g *gateway destination*

admroute -o delete [-p] [-t net | host] -g *gateway destination*

admroute -o list [-qv] [all | *destination* ...]

admroute -o search -g *gateway destination*

DESCRIPTION

admroute manages the current and permanent routing databases. The current routes are the routes in the network routing table that are used by the kernel to make routing decisions. The permanent routes are routes that are stored in a file and installed as current routes when the network is started. When the network is started, admroute manages the current routes, and optionally the permanent routes. When the network is stopped, admroute manages only the permanent routes.

Each route consists of: *type*, *destination*, *gateway*, and *metric*.

Operations

add — Add a new route to the current and/or permanent routing database.

modify — Change a route from the current and/or permanent routing database. The destination *type*, *destination*, *gateway*, and *metric* may all be changed.

delete — Remove a route from the current and/or permanent routing database.

list — List one or more routes in the current or permanent routing database. Information is listed about all routes to the *destinations*; if no *destinations* are given or if *destination* is all, information about all routes is listed.

search — Search for the *destination/gateway* combination in the current or permanent routing database. If the route is found, it is echoed to stdout with a single space delimiter between each field. Otherwise, exit with an error code.

Options

-p — Make the requested operation permanent so that the permanent routes are updated in addtion to the current routes. The routing entry is updated in the permanent routing database located in the tcpip.params(4) file.

-t net | host

net | host is the type of destination for the route. When the network is subnetted, this parameter is required to distinguish a network address from a host address. If the address is not on a subnetted network, then it can be determined from the address whether it represents a host or a network, and the parameter is not required.

-g *gateway*

gateway is the symbolic name or Internet address to route packets through in order to reach the *destination*. If a symbolic name is specified, then an entry will be added to local hosts(4) database.

-m *metric* *metric* is the number of network hops to reach the *gateway*. Currently, the only valid values are 0 and 1. A value of 0 indicates that the route goes through a local interface to the directly-connected network. A value of 1 indicates that the route goes through a gateway. The default is 1.

-d *new-destination*
new-destination is the symbolic name or Internet address that will replace *destination* in the `modify` command. If a symbolic name is specified, then an entry will added to the local `hosts` or `networks`(4) database.

-G *new-gateway*
new-gateway is the new name or Internet address that replaces *gateway*. If a symbolic name is specified, then an entry will added to the local `hosts` database. Without this option the *gateway* is not changed.

-q "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-v "Verbose." Produce a formatted listing with headers and aligned columns. This option is enabled by default.

`admroute` expects all Internet addresses to be of the form

a.b.c.d,

where *a* is a decimal number between 0 and 224, and *b*, *c*, and *d* are decimal numbers between 0 and 255.

FILES

/etc/tcpip.params
TCP/IP parameters file contains the permanent routes database.

OUTPUT

The `list` and `search` operations write their output to stdout.

The verbose form of the `list` operation outputs information in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:
type destination gateway metric

This format is also used by the `search` operation.

DIAGNOSTICS

Warnings

- The `delete` operation is requested, and *destination/gateway* does not exist.

Errors

- The `add` operation is requested, and the *destination/gateway* already exists.
- The `modify` operation is requested, and the *new-destination/new-gateway* already exists.
- The `modify` or `search` operation is requested, and *destination/gateway* does not exist.
- The *destination*, *new-destination*, *gateway*, or *new-gateway* parameter is specified as an invalid Internet address.

- The *destination*, *new-destination*, *gateway*, or *new-gateway* is specified as a symbolic name that cannot be resolved to a Internet address by the `hosts` or `networks` database.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

4 A symbolic name could not be resolved, or an Invalid Internet address was entered.

SEE ALSO

`netstat`(1C), `route`(1M), `hosts`(4), `networks`(4), `tcpip.params`(4).

NOTES

Only the system administrator may modify the routing databases.

NAME

admrshell - manage the remote and restricted shell names

SYNOPSIS

```
admrshell -o set sysV | bsd

admrshell -o get
```

DESCRIPTION

Admrshell manages the remote and restricted shell names. The shell names may be either compatible with AT&T System V or BSD.

Operations

set — Set the remote and restricted shell names to be compatible with with AT&T System V or BSD.

get — Get whether the remote and restricted shell names are compatible with with AT&T System V or BSD.

Options

None.

FILES

/usr/bin/remsh
: Remote shell executable file.

/usr/bin/restsh
: Restricted shell executable file.

/usr/bin/rsh
: Link to remote or restricted shell depending on which system has been chosen. Linked to /usr/bin/restsh if sysV compatible. Linked to /usr/bin/remsh if bsd compatible.

OUTPUT

The get operation writes the system compatibility to stdout.

DIAGNOSTICS

Warnings

None.

Errors

- The *system* is not either sysV or bsd.
- The remote and restricted shell files and links are not as expected because they have been manually changed.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Access to the set operation is based on write access to /usr/bin. Generally, only the system administrator has this access. Access to the get operation is based on read access to /usr/bin. Generally, all have this access.

SEE ALSO

sh(1), remsh(1C), restsh(1C), rsh(1C).

NAME

admsar - manage system activity monitoring and reporting

SYNOPSIS

admsar -o start [-e] [-t *interval*] [-n *samples*] [*name*]

admsar -o stop *name* ...

admsar -o delete *name* ...

admsar -o list [-qv] [-r *report-type*] [*name*]

admsar -o list [-qv] [-r *report-type*] -t *interval* -n *samples*

DESCRIPTION

The admsar command starts or stops system activity monitoring, or produces a report from a previous (or current) monitoring session. System activity monitoring and reporting are performed using the sar(1) command. Refer to that manual entry for more information.

Operations

start — Initiate a monitoring session. A sampling interval (in seconds) and the number of samples are specified. The monitoring session will stop automatically after the requisite number of samples are taken. The results of the sampling are placed in the named data file (if no name is specified, one will be created based on the current day of the month).

stop — Terminate a monitoring session. This is provided as an alternative to allowing the monitoring session to cease of its own accord. The data collection associated with the named data files is stopped, or all data collection is stopped if *name* is all.

delete — Remove specified named data files, or all data files if *name* is all.

list — Produce one of several system activity reports, as selected with the -r option. The report can use the sampling information in a named data file created by the start operation. If an interval or sample count is specified, the sample information is collected directly from the system instead of from a data file (and the command waits for sampling to be completed).

Options

-e — Erase the data file before writing to it. Otherwise, new data is appended to the file.

-t *interval*
The number of seconds between samples. The default interval is 5 seconds.

-n *samples*
The number of samples to take. The default number of samples is 10.

-r *report-type*
The type of report. *report-type* is a single letter selected from the option letters allowed for sar(1). An additional report type is F, which lists all known data files; no data file name, interval, or sample count should be specified with this report type. If no report type is specified, type u (user, system, and idle times) is used.

-q — Quiet. Headings are left off of reports.

`-v` Verbose. This option is enabled by default.

EXAMPLES

Suppose you want to initiate a one-minute monitoring session with samples every five seconds, and you don't want to include any previous sampling that may have been done earlier in the day. Use:

```
admsar -o start -e -t 5 -n 12
```

After a minute, you may then produce reports based on this sampling. Suppose you want a report on the system calls during that session, and then a report on the CPU usage. Use:

```
admsar -o list -r c
admsar -o list -r u
```

If you want to keep this sample separate from other samples you might be taking, use a named data file:

```
admsar -o start -t 5 -n 12 midday
admsar -o list -r w midday
```

Once all reports have been produced, you may remove all sample data files by using:

```
admsar -o delete all
```

FILES

/var/adm/sa/spd.Day*dd*
: Default sampling data file, where *dd* is the two-digit day of the month. All activity for the current day is stored together in this file.

/var/adm/sa/spd.*name*
: Named sampling data file.

OUTPUT

The `list` command produces the output from the `sar`(1) command, as described in that manual entry.

DIAGNOSTICS

Warning

None.

Errors

- A `list` operation was attempted and the named report file does not exist.
- A report file name, sampling interval, or sample count was specified with the `list` operation and the `F` report type.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful. The system activity monitor has not yet been run (`list`).

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`sar`(1), `sar`(1M), `sysadm`(1M).

NOTES

Only the super-user may use the `start`, `stop`, and `delete` operations.

Having more than one monitoring session active for the same data collection file name will produce unpredictable results.

NAME

admservice - manage service database

SYNOPSIS

admservice -o add [-y] [-l *alias-list*] -n *port-number* -p *protocol service-name*

admservice -o delete [-y] -p *protocol service-name*

admservice -o modify [-y] [-s *new-service-name*] [-P *new-protocol*] [-n *port-number*] [-l *new-alias-list*] -p *protocol service-name*

admservice -o list [-y] [-qv] [all | *service-name* ...]

admservice -o search [-y] -p *protocol service-name*

DESCRIPTION

admservice manages the local or the NIS (YP) services(4) database. Each services database entry consists of: *service-name*, *protocol*, *port-number*, and a list of aliases.

Operations

add — Add a new service entry into the services database.

modify — Change a service entry from the services database. The *service-name*, *protocol*, *port-number*, and *alias-list* may be changed.

delete — Remove a service entry from the services database.

list — List one or more service entries from the services database. Information is listed about each of the *service-name*s; if no *service-name*s are given or if *service-name* is all, information about all services is listed.

search — Search for a service entry from the services database. If the entry is found, it is echoed to stdout. If the entry is not found, the operation exits with an error code.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. If specified with the add, delete, or modify operations, this option is valid only if the machine on which the command is run is the NIS master. The -y option uses the default domain name derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-p *protocol*

protocol is the transport protocol for this service(e.g. tcp, udp). The *protocol* must be found in the protocols(4) database.

-P *new-protocol*

new-protocol is the new transport protocol that replaces *protocol* in the modify operation. Without this option the current *protocol* is preserved.

-n *port-number*

port-number is the well-known port number assigned for this service. If this option is omitted in the modify operation, the current *port-number* is preserved.

-l *alias-list*

alias-list is a list of aliases by which the *service* can be referenced. If this option is omitted in the modify operation, the current *alias-list* is

preserved.

`-s` *new-service*
: *new-protocol* is the new service name that replaces *service-name* in the `modify` operation. Without this option the current *service* is preserved.

`-q`
: "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

`-v`
: "Verbose." Produce a formatted listing with headers and aligned columns. This is enabled by default.

FILES

/etc/services
: Local `services` database file.

OUTPUT

The `list` and `search` operations write their output to stdout. The verbose form of the `list` operation outputs information in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

service_name protocol port_number alias1 alian2 aliasN ...

This format is also used by the `search` operation.

DIAGNOSTICS

Warnings

- The `delete` operation is requested, and *service-name/protocol* does not exist.

Errors

- The `-y` option is specified for an `add`, `delete`, or `modify` operation and the host is not the NIS master.
- The `add` operation is requested, and *service-name/protocol* already exists.
- The `modify` operation is requested, and *new-service-name/new-protocol* already exists.
- The `modify` or `search` operation is requested, and *service-name/protocol* does not exist.
- The `-y` option is given, and there is an error exporting NIS maps.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line. does not exist.

NOTES

Access to the `add`, `delete`, and `modify` operations is based on write permissions on the `services` database file. Generally, only the system administrator has this access. Access to the `list` and `search` operations is based on read permissions on the `services` database file. Generally, all have this access.

SEE ALSO

`domainname`(1), `yp`(3R), `protocols`(4), `services`(4).

NAME

admsnmpcommunity - manage the SNMP community database

SYNOPSIS

admsnmpcommunity -o add -a *access* -h *host*|any *community*

admsnmpcommunity -o delete [-h *host*|any] *community*

admsnmpcommunity -o modify [-A *new-access*] [-C *new-community*] [-H *new-host*|any] [-h *host*|any] *community*

admsnmpcommunity -o list [-q|v] [-h *host*|any] [all|*community* ...]

admsnmpcommunity -o search [-h *host*|any] *community*

DESCRIPTION

Use admsnmpcommunity to manage the SNMP community database. The community database is located in the /etc/snmpd.communities file. Each entry in the file defines a community that the SNMP agent, snmpd, will recognize. Each entry consists of a *community* name, *host* name, and a level of *access*. Specify the *community* and *host* fields so that they uniquely identify the entry.

add — Add a new entry to the database. Specify the *access* as one of the following: read, write, or none. Specify the *host* argument as either a hostname, Internet address, or the keyword any. The keyword any indicates that the Internet address 0.0.0.0 should be used, which allows all hosts to use this *community* name.

delete — Remove the entry for *community* from the database. If multiple entries for *community* exist, and the -h option is not used to qualify which entry to delete, all matching entries are deleted.

modify — Change the existing entry for *community* in the database. If multiple entries for *community* exist, you should use the -h option to qualify which entry to modify.

list — List the entry for *community* from the database. If no *community* is specified or the keyword all is given, all entries in the database are displayed.

search — Search for the *community* in the database. If the entry is found, it is written to stdout, otherwise the operation exits with an error code.

Options

-h *host*|any
: Use the *host* option to qualify the community on which to perform the operation should there be multiple entries with the same *community* name. *host* may be specifed as the keyword any to indicate the wildcard address 0.0.0.0, which means any Internet address.

-A *new-access*
: Use the *new-access* option to specify the access level to be assigned to the *community* after the modification is complete. *new-access* should be specified as one of the following: read, write, or none.

-a *access*
: Use the *access* option to specify the access level to be assigned to the *community*. *access* should be specified as one of the following: read, write, or none.

-C *new-community*
: Use the *new-community* option to specify the community name to be assigned to the *community* after the modification is complete.

-H *new-host*
Use the *new-host* option to specify the host name to be assigned to the *community* after the modification is complete.

-q Specify the quiet option to produce an unformatted listing (that is, no headers).

-v Use the verbose option to produce a formatted listing with headers and aligned columns. This option is enabled by default.

FILES

/etc/snmpd.communities the communities database.

DIAGNOSTICS

Warnings

- If the agent can not be sent a SIGHUP signal to request reconfiguration.
- If the `delete` or `modify` operation is requested and the *community-host* pair does not exist.

Errors

- If the `add` operation is requested and the *community-host* pair already exists.
- If the `add` operation is requested and the *access* is not one of the following: `read`, `write`, or `none`.
- If the `modify` operation is requested and the *new-access* is not one of the following: `read`, `write`, or `none`.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Access to the `add`, `delete`, and `modify` operations is granted based on write access to the `/etc/snmpd.communities` file. Generally, only the system administrator has this access. Access to the `list` and `search` operation is granted based upon read access to the `/etc/snmpd.communities` file. Generally, everyone has this access.

SEE ALSO

`snmpd`(1M), `snmpd.communities`(4), `sysadm`(1M).

NAME

admsnmpobject - manage the snmpd object database

SYNOPSIS

admsnmpobject -o get [-q|v] *object*

admsnmpobject -o set [-a *value*] *object*

admsnmpobject -o reset *object*

admsnmpobject -o list [-q|v] [all|[*object* ...]]]

admsnmpobject -o supported [-q|v] [all|[*object* ...]]]

DESCRIPTION

Use admsnmpobject to manage the MIB object database. The MIB object database is located in the /etc/snmpd.config file. This database contains optional entries to specify values for MIB objects that are machine dependent. Use this command to tailor the objects in the database appropriately for your system. Each entry in the object database consists of an objectname, an equal sign, and a value. Not all objects maintained by snmpd can be modified. Use the supported operation to see a list of objects that you can modify.

Most adminstrators only need to specify the sysContact and sysLocation objects because the agent can provide reasonable defaults for all other supported objects.

get Get the current value for the requested *object*. If the *object* is in the database the value is printed, otherwise, the default value is printed.

set Set the *object* to the requested *value*.

reset Reset the value of the *object* to the system supplied default. This option removes the entry for *object* from the database.

list List the current value for *object*.

supported
List the *objects* that are allowed in the database and give the default values that are assigned to them.

Options

-a *value* Use the *value* option to specify the value to be assigned to the requested *object*.

-q Use the quiet option to produce an unformatted listing.

-v Use the verbose option to produce a formatted listing of object value pairs with headers and aligned columns.

OUTPUT

All operations write their output to stdout and diagnostics to stderr.

If you specify the -v "verbose" option with the list, get, or supported operations, the object-value pairs are displayed in aligned columns with headers.

If you specify the -q "quiet" option with the list or get operation, headers are suppressed and only the values for the objects are written. If you specify the -q "quiet" option with the supported operation, headers are suppressed and only valid object names are written. If you do not specify either the -q or -v option with the list, get, or supported operations, headers are suppressed, and the output is in an object=value format.

FILES

/etc/snmpd.config the MIB object database.

DIAGNOSTICS

Warnings

- If the agent can not be sent a SIGHUP signal to request reconfiguration.

Errors

- The *object* requested is not one of the supported objects.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`snmpd`(1M), `snmpd.config`(4), `sysadm`(1M).

NOTES

Access to the `get`, `set`, and `reset` operations is granted based upon write access to the `/etc/snmpd.config` file. Generally, only the system administrator has this access. Access to the `list` and `supported` operation is granted based upon read access to the `/etc/snmpd.config` file. Generally, everyone has this access.

NAME

admsnmptrap - manage the SNMP traps database

SYNOPSIS

admsnmptrap -o add -p *port*|default -c *community host*

admsnmptrap -o delete [-c *community*] *host*

admsnmptrap -o modify [-P *new-port*|default] [-C *new-community*] [-H *new-host*] [-c *community*] *host*

admsnmptrap -o list [-q|v] [-c *community*] [all|*host* ...]

admsnmptrap -o search [-c *community*] *host*

DESCRIPTION

Use admsnmptrap to manage the SNMP traps database. The traps database is located in the /etc/snmpd.trap_communities file. The SNMP agent, snmpd, will send traps to the hosts specified in the traps database. Each entry consists of a *community* name, *host* name, and a *port* number. Specify the *community* and *host* pair so that they uniquely identify an entry in the file.

add — Add a new entry to the database. Specify the *port* as either an integer value or the keyword default, which will set the port number to 162. Specify the *community* as an ASCII string of 64 or fewer characters. Specify the *host* as either a hostname or Internet address indicating where the agent should send the traps.

delete — Remove the entry for *host* from the database. If multiple entries for *host* exist, and the -c option is not used to qualify which entry to delete, all matching entries are deleted.

modify — Change the existing entry for *host* in the database. If multiple entries for *host* exist, you should use the -c option to qualify which entry to modify.

list — List the entry for *host* from the database. If no *host* is specified or the keyword all is used then all the entries in the database are displayed.

search — Search for an entry for *host* in the database. If the entry is found, it is written to stdout, otherwise the operation exits with an error code.

Options

-c *community*
You can use the *community* option to qualify the *host* on which to perform the operation should there be multiple entries for the *host* in the database.

-P *new-port*
Use the *new-port* option to specify the port number to be assigned to the *host* entry after the modification is complete.

-p *port* — Use the *port* option to specify the port number to be assigned to the *host* entry. You can specify the keyword default which will set the port number to 162.

-C *new-community*
Use the *new-community* option to specify the community name to be assigned to the *host* entry after the modification is complete.

-H *new-host*
Use the *new-host* option to specify the host name to be assigned to the existing *host* entry after the modification is complete.

`-q` Use the "Quiet" option to produce an unformatted listing (that is, no headers)

`-v` Use the "Verbose" option to produce a formatted listing with headers and aligned columns. This is the default.

FILES

/etc/snmpd.trap_communities the traps database.

DIAGNOSTICS

Warnings

- If the agent can not be sent a SIGHUP signal to request reconfiguration.
- If you request the `delete` or `modify` operation and the *community-host* pair does not exist.

Errors

- If you request the `add` operation and the *community-host* pair already exists.
- If you request the `modify` operation and the *community-host* pair is not unique.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`snmpd`(1M), `snmpd.trap_communities`(4), `sysadm`(1M).

NOTES

Access to the `add`, `delete`, and `modify` operations is granted based upon write access to the `/etc/snmpd.trap_communities` file. Generally, only the system administrator has this access. Access to the `list` and `search` operation is granted based upon read access to the `/etc/snmpd.trap_communities` file. Generally, everyone has this access.

NAME

admsvcorder - manage search order for /etc/hosts, NIS, and DNS databases

SYNOPSIS

admsvcorder -o set -1 *first-database* [-2 *second-database* [-3 *third-database*]]

admsvcorder -o get [-qv]

DESCRIPTION

Admsvcorder manages the order that the /etc/hosts, Network Information Service, and Domain Name System resolver databases are searched for hostname-to-address resolution.

Operations

set — Set the order that the databases are searched.

get — Get the order that the databases are searched.

Options

-1 *first-database*
: *first-database* is the database that will be searched first in hostname-to-address resolution.

-2 *second-database*
: *second-database* is the database that will be searched second in hostname-to-address resolution. If unspecified then only the first database will be searched.

-3 *third-database*
: *third-database* is the database that will be searched third in hostname-to-address resolution. If unspecified then only the first and second databases will be searched.

-q
: "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-v
: "Verbose." Produce a formatted listing with headers and aligned columns. This is the default.

admsvcorder expects the databases to be specified with the following convention:

NIS or nis for the Network Information Service (formerly Yellow Pages) database.

EHOSTS or ehosts for the local /etc/hosts database.

RES or res for the Domain Name System resolver.

FILES

/etc/svcorder
: Search order file.

OUTPUT

The get operation writes its output to stdout.

The verbose form of the operation outputs information in aligned columns with column headers.

The quiet form of the operation suppresses headers and each database is printed in the order that it is searched on separate lines.

DIAGNOSTICS

Warnings

None.

Errors

- The `set` operation is requested, and the *−1* option is not specified.
- The `set` operation is requested, and the *−3* option is specified, but the *−2* option is not specified.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line. does not exist.

SEE ALSO

`resolver`(3), `YP`(3R), `hosts`(4), `svcorder`(4).

NOTES

Access to the `set` operation is granted based upon the write access to /etc/svcorder. Generally, only the system administrator has this access. Access to the `get` operation is granted based upon the read access to /etc/svcorder. Generally, all have this access.

NAME

admswap - manage swap areas

SYNOPSIS

admswap -o add *dev-file ...*

admswap -o delete *dev-file ...*

admswap -o list [-qv] [*dev-file ...*]

DESCRIPTION

The admswap command is used to manage system swap areas. The argument, *dev-file*, specifies a block special device file (e.g. logical disk). If *dev-file* is not an absolute path, it is assumed to be the name of a device file in the /dev/dsk directory.

Operations

add — Add a swap entry for *dev-file* to the fstab file and call swapon(1M) to begin using it as a swap area.

delete — Delete the swap entry for *dev-file* from the fstab file. Note, however, that *dev-file* will continue to be used as a swap area until the system is rebooted.

list — List swap areas to stdout.

Options

-q — Quiet. Produce an unformatted listing (i.e. no headers).

-v — Verbose. Produce a formatted listing with headers and aligned columns. This option is enabled by default.

EXAMPLE

The following command line would be used to add the logical disk /dev/dsk/swap2 to fstab and begin using it as a swap area:

```
admswap -o add swap2
```

OUTPUT

The list operation produces a list of swap areas consisting of a header followed by the names of the swap devices each on separate lines. The -q option suppresses the printing of the header.

FILES

/etc/fstab — file system table

DIAGNOSTICS

Warnings

None.

Errors

- *dev-file* does not reference a valid swap device.
- Attempt is made to add a swap area entry that already exists or delete a swap area entry that doesn't exist.
- An attempt to begin swapping on a new swap area fails.

Exit Codes

0 — The operation was successful.

1 — The operation was unsuccessful.

2 — The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Super-user privilege is required for the `add` and `delete` operations.

SEE ALSO

`swapon`(1M), `sysadm`(1M), `fstab`(4).

NAME

admtape - manipulate the default parameters for tapes

SYNOPSIS

```
admtape -o get { -f | -M }
admtape -o set { -f | -M } value
admtape -o list [ -qv ] { -f | -M }
```

DESCRIPTION

Admtape gets or sets the current default parameters for tape operations or lists the possible values for the parameters. Currently the list of parameters includes the default file, and the default medium type.

The default file is a device name, such as /dev/rmt/0, which will be given as the default drive for tape operations, including performing backups and loading software.

The default medium is a tag from the dumptab file (see dumptab(4)) which is the default medium to use for backups.

Operations

get	Write the default value for a given parameter to the standard output.
list	List the possible values for the given parameter to standard output.
set	Set the system-wide default for the given parameter to *value*.

Options

-f	Manipulate the default tape drive for the system.
-M	Manipulate the default medium type for the system.
-q	Quiet. Omit header lines from the output of the list operation.
-v	Verbose. Include header lines in the output of the list operation. This option is enabled by default.

One of -f and -M must be given to specify which tape parameter is used for the operation.

EXAMPLE

In order to determine the default tape drive for the system, the following command line is used:

```
$ admtape -o get -f
/dev/rmt/0
$
```

To set the default medium type, this command line may be used:

```
$ admtape -o set -M cartridge
$
```

FILES

/etc/sysadm/mt
: The file used to maintain the tape defaults information.

/etc/dumptab
: List of possible media names.

OUTPUT

The `list` operation displays the list of possible values for the given parameter to standard output. No other information is provided.

DIAGNOSTICS

Warnings

- The requested device file does not exist.
- The requested medium type is not found in the dumptab file.

Errors

None.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`admbackup`(1M), `admdefault`(1M), `admpackage`(1M), `dump2`(1M), `dumptab`(1M), `sysadm`(1M).

NOTES

This command replaces the `osysadm`(1M) *tapedefaults* command.

You must have write permission to the `/etc/sysadm/mt` file to use the `set` operation. Usually, only the super-user has such permission.

The `get` and `set` operations manipulate the `/etc/sysadm/mt` database. Support for this database will be removed in a future release. The information will be maintained by `admdefault`(1M) instead.

NAME

admtcpipdaemon - manage the TCP/IP servers

SYNOPSIS

admtcpipdaemon -o add [-I -s *service* -p *protocol*] [-a *args*] *server*

admtcpipdaemon -o delete [-I] *server* ...

admtcpipdaemon -o modify [-I [-s *service*] [-p *protocol*]] [-a *args*] [-d *new-server*] *server*

admtcpipdaemon -o list [-I] [-qv] [all | *server* ...]

admtcpipdaemon -o start *server* ...

admtcpipdaemon -o stop *server* ...

DESCRIPTION

Admtcpipdaemon manages the TCP/IP servers (daemons). The TCP/IP servers are split into two databases: independent servers that run as their own process, and inetd(1M) servers that are spawned off by the inetd server when their services are requested.

The independent servers database consists of a list of server programs and arguments to the program. Each of these servers will be started when the TCP/IP network is started.

The inetd servers database consists of a list of server programs, arguments to the program, the *service* implemented by the server, and the *protocol* used by the server. The servers will be spawned off by the inetd server if the inetd server is in the database of independent servers.

Operations

add — Add a server to the independent or inetd servers database.

delete — Remove one or more servers from the independent or inetd servers database.

modify — Change a server from the independent or inetd servers database.

list — List one or more servers from the independent or inetd servers database.

start — Start or restart one or more servers from the independent servers database.

stop — Stop one or more servers from the independent servers database.

Options

-I — Perform the requested operation on the inetd servers database. Without this option the operation is performed on the independent servers database. This option is not valid for the start or stop operations, which operate only on the independent servers database.

-s *service* — *service* is the network service implemented by the server in the inetd servers database. This *service* must be found in the services(4) database. This option is only valid when operating on the inetd servers database.

-p *protocol* — *protocol* is the transport protocol used by the server in the inetd servers database. This *protocol* must be found in the protocols(4) database. This option is only valid when operating on the inetd servers database.

`-a` *args* *args* are the command line arguments to the server program.

`-d` *new-server*
new-server is the new server name that replaces *server* in the modify operation. Without this option the current server program will be preserved.

`-q` "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

`-v` "Verbose." Produce a formatted listing with headers and aligned columns. This option is enabled by default.

EXAMPLES

In the following examples, the system administrator performs the following actions: add the *ftpd* server to the `inetd` server database; modify the *ftpd* server, changing the arguments to *-d -l*; add the *smtp* server, to the independent server database with *-q30m* as its argument.

```
admtcpipdaemon -o add -I -s ftp -p tcp ftpd
admtcpipdaemon -o modify -I -a "-d -l" ftpd
admtcpipdaemon -o add -a "-q30m" smtp
```

FILES

/etc/tcpip.params
File that contains the independent servers database.

/etc/inetd.conf
File that contains the inetd servers database.

OUTPUT

The `list` operation writes its output to stdout.

The verbose form of the `list` operation outputs the entry in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space.

For `inetd` servers the fields are in the following order:
service protocol server args

For independent servers the fields are in the following order:
server args

DIAGNOSTICS

Warnings

- The `delete`, `start`, or `stop` operation is requested, and *server* does not exist.

Errors

- The `add` operation is requested, and *server* already exists.
- The `modify` operation is requested, and *server* does not exist.
- The `modify` operation is requested, and *new-server* already exists.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

NOTES

Access to the `add`, `delete`, and `modify` operations is granted based on write permissions for the given servers database file. Generally, only the super-user may perform these operations. Access to the `list` operation is granted based on read permissions for the given servers database file. Only the super-user may perform the `start` and `stop` operations.

SEE ALSO

`inetd`(1M), `inetd.conf`(4), `protocols`(4), `services`(4) `tcpip.params`(4).

NAME

admtcpipparams - manage the TCP/IP host parameters

SYNOPSIS

admtcpipparams -o set [-n *hostname*] [-i *hostid*]

admtcpipparams -o get [-qv]

DESCRIPTION

Admtcpipparams manages the TCP/IP host parameters in the tcpip.params(4) database. The parameters include hostname(1C) and hostid(1C).

Operations

set — Set the hostname and/or the hostid. The hostname and hostid are set currently and will be set when the network is started.

get — Get the hostname and the hostid.

Options

-n *hostname*
hostname is the name that uniquely identifies a host system. Every host has a hostname for each network interface which associates an Internet address with that interface. But the *hostname* uniquely identifies a host system. By convention, the *hostname* is also the hostname for the primary network interface.

-i *hostid* *hostid* is the hexadecimal number that uniquely identifies a host system. By convention, the *hostid* is the primary network interface's Internet address in hexadecimal form.

-q "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-v "Verbose." Produce a formatted listing with headers and aligned columns. This option is enabled by default.

FILES

/etc/tcpip.params
TCP/IP parameters file which stores the TCP/IP host parameters.

OUTPUT

The get operation writes its output to stdout.

The verbose form of the get operation outputs information in aligned columns with column headers.

The quiet form of the operation suppresses headers and the *hostname* and *hostid* are printed out with one space delimiter.

DIAGNOSTICS

Warnings

None.

Errors

- The *hostid* is not in hexadecimal form.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`hostname`(1C), `hostid`(1C), `tcpip.params`(4).

NOTES

Only the system administrator has access to the set operation. Access to the `get` operation is granted based upon read access to the `tcpip.params` file. Generally, all have read access to the `tcpip.params` file.

NAME

admterminal - manage terminal ports

SYNOPSIS

admterminal -o add [-a *alternate-prompt*] [-d] [-i *disabled-msg*] [-l *tty-label*] [-p *pm-tag*] [-s *service*] [-t *term-variable*] [-y *comment*] *tty-num* ...

admterminal -o delete [-p *pm-tag*] *tty-num* ...

admterminal -o modify [-a *alternate-prompt*] [-d] [-i *disabled-msg*] [-l *tty-label*] [-p *pm-tag*] [-s *service*] [-t *term-variable*] [-y *comment*] *tty-num*

admterminal -o disable [-p *pm-tag*] *tty-num* ...

admterminal -o enable [-p *pm-tag*] *tty-num* ...

admterminal -o list [-qvx] [-p *pm-tag*] *tty-num* ...

DESCRIPTION

Admterminal manages terminal ports monitored by ttymon(1M) port monitors under control of the Service Access Facility. It provides access to the most commonly-used terminal management options provided by ttymon and is intended as a simple alternative to the more complex admportservice(1M) and pmadm(1M) commands.

Operations

add — Add one or more new terminals to be monitored by a ttymon port monitor. If no ttymon port monitors exist, the default port monitor, ttymon1, will also be added as part of this operation.

delete — Remove one or more terminals from those being monitored by a ttymon port monitor.

disable — Disable one or more terminals. A disabled terminal will not accept user logins.

enable — Enable one or more disabled terminals.

modify — Change attributes of a terminal.

list — List information about one or more terminals. If a specific ttymon port monitor is specified via -p *pm-tag*, the listing is restricted to those terminals monitored by that port monitor.

Options

-a *alternate-prompt*
Alternate login prompt for the terminal. The default login prompt is "login: ".

-d
Disabled. If this option is specified, the terminal(s) will be disabled when added. Terminals are enabled by default.

-i *disabled-msg*
Disabled message. This is the message that will be displayed on a terminal when it is disabled. The default is to display no message.

-l *tty-label*
The name of an ttydefs(4) entry which specifies the line speed and other terminal settings. The default *tty-label* is "9600".

`-p` *pm-tag* Port monitor tag. The name of the controlling `ttymon` port monitor for the terminal(s). This option is required only if the system has more than one `ttymon` port monitor. If only one `ttymon` port monitor is present on the system, it is used by default. If this option is specified for the `list` operation, the listing will be restricted to those terminals monitored by the specified port monitor.

`-q` "Quiet." List minimal information. Only the device file, port monitor, tty label, state and comment for each terminal are listed.

`-s` *service* Login service for the terminal(s). This is the program that will be called when a user attempts to login to the terminal(s). The default login service is `/usr/bin/login`.

`-t` *term-variable*
The value to which the TERM environment variable should be set before invoking the login service. If this option is not specified, the TERM variable will not be set.

`-y` *comment*
Comment associated with the terminal.

`-v` "Verbose." Produce a detailed, formatted listing. This is the default output format.

`-x` "Extended." List the TERM variable setting in addition to the other information about each terminal. Use of this option will degrade the performance of the `list` command because the TERM variable must be retrieved from a separate file for each terminal.

Output

The "verbose" format of the `list` operation reports the following information to stdout: terminal device file, port monitor, tty label, state, service, prompt, disabled message and comments. If `-x` is specified, the value of the TERM variable will be listed before the comments. The information is presented with labels and indentation suitable for easy viewing.

The "quiet" output format reports the terminal device file, port monitor, tty label, state and comments for each terminal. If `-x` is specified, the value of the TERM variable will be displayed as "`TERM=`*value*" before the comments field. Information for each terminal is printed on a single line and each field is delimited by spaces. The comments field is preceded by a pound sign (#).

FILES

/etc/ttydefs
File which contains entries for defining line speed and other terminal settings.

DIAGNOSTICS

Errors

`admterminal` uses `ttyadm(1M)` and `pmadm(1M)` to perform the requested operation for the terminal(s). It reports any error conditions returned by these commands.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`login`(1), `admportmonitor`(1M), `admportservice`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `ttyadm`(1M), `ttymon`(1M), `ttydefs`(4), *Managing the DG/UX System*.

NOTES

All operations except for `list` require superuser access.

NAME

admtrustedhost - manage the trusted hosts database

SYNOPSIS

admtrustedhost -o add [-u *user*] *trusted-host*

admtrustedhost -o delete [-u *user*] *trusted-host*

admtrustedhost -o modify [-U *new-user*] [-h *new-trusted-host*] [-u *user*] *trusted-host*

admtrustedhost -o list [-qv] [*trusted-host* ...]

admtrustedhost -o search [-u *user*] *trusted-host*

DESCRIPTION

admtrustedhost manages the trusted hosts database. The trusted hosts database is located in the hosts.equiv(4) file. Each trusted host entry consists of a host name, and a user name(with a passwd(4) entry) or all. The host name may be a fully-qualified name or a simple name. If both fully-qualified and simple host names are being used, then the entries must be operated on separately. Each trusted host entry allows *user* on *trusted-host* remote access for rlogin(1C) and remsh(1C).

add — Add a new *trusted-host/user* pair to the trusted hosts database.

modify — Change a trusted host entry from the trusted hosts database. The *trusted-host* and/or *user* may be changed.

delete — Remove a trusted host entry from the trusted hosts database.

list — List one or more *trusted-hosts* and their *users* from the trusted hosts database. Information is listed about each of the *trusted-hosts*; if no *trusted-hosts* are given or *trusted-host* is all, information about all trusted hosts is listed.

search — Search for the *trusted-host/user* pair in the trusted hosts database. If the pair is found, it is echoed to stdout with a single space delimiter between each field. If the pair is not found, the operation exits with an error code.

Options

-u *user* — *user* is the user on the *trusted-host* allowed remote access. If -u is omitted or if *user* is all, then all users from *trusted-host* are allowed remote access. The superuser is not allowed remote access via hosts.equiv.

-U *new-user*
new-user is the user that will replace *user* in the modify command. Without this option the *user* is not changed.

-h *new-trusted-host*
new-trusted-host is the trusted host that will replace *trusted-host* in the modify command. Without this option the *trusted-host* is not changed.

-q — "Quiet." Produce an unformatted listing (i.e. no headers, fields delimited by a single space).

-v — "Verbose." Produce a formatted listing with headers and aligned columns. This option is enabled by default.

OUTPUT

The list and search operations write their output to stdout.

The verbose form of the list operation outputs information in aligned columns with column headers.

If `-q` option is specified with the `list` operation, headers are suppressed and each entry is printed on a separate line. The fields within the entry are delimited by a single space, and are in the following order:

trusted_host user

This format is also used by the `search` operation.

FILES

/etc/hosts.equiv

Trusted hosts database file.

DIAGNOSTICS

Warnings

- The `delete` operation is requested, and *trusted-host/user* does not exist.

Errors

- The `add` operation is requested, and *trusted-host/user* already exists.
- The `modify` operation is requested, and *new-trusted-host/new-user* already exists.
- The `modify` or `search` operation is requested, and *trusted-host/user* does not exist.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`remsh`(1C), `rlogin`(1C), `hosts.equiv`(4).

NOTES

Access to the `add`, `delete`, and `modify` operations is granted based upon write access to the `hosts.equiv` file. Generally, only the system administrator has this access. Access to the `list` and `search` operation is granted based upon read access to the `hosts.equiv` file. Generally, all users have this access.

NAME

admuser – manage user information in the password database

SYNOPSIS

admuser -o add [-yp] [-u *uid*] [-g *gid*] [-d *home-directory*] [-m] [-c *comment*] [-s *shell*] *login*

admuser -o modify [-yp] [-u *uid*] [-g *gid*] [-d *home-directory*] [-c *comment*] [-s *shell*] [-l *new-login*] *login*

admuser -o delete [-yr] *login* ...

admuser -o list [-yqva] [-u *uid*] [-g *gid*] [*login* ...]

admuser -o get [-qv]

admuser -o set [-g *gid*] [-b *base-directory*] [-s *shell*] [-k *skeleton-directory*]

DESCRIPTION

admuser will manage information in the local or NIS (YP) password database file. The passwd(4) file contains basic information about each user's account.

admuser is normally run by the system administrator on the NIS master machine if the system is running NIS, or on any host if the system is not running NIS.

Operations

add — This operation will add a new user to the password file. If the specified *login* already exists, the operation will not be successful. If the -m option is specified then the *home-directory* for *login* will be created and the contents of the *skeleton-directory* will be copied to the home directory. Only the superuser may execute this operation.

modify — This operation will modify the currently existing information in the local or NIS password file. The command-line options will determine the changes that are made to the *login* entry in the password database. Only the superuser may execute this operation.

delete — This operation will delete the given *login*(s) from the password file. If the -r option is specified, then the user's home directory will be removed from the system. Only the superuser may execute this operation.

list — This operation will list the *login*(s) in the password database who match the qualifications specified in the command-line options. If no command-line options are given, then only the local user database will be listed. Any user may execute this operation.

get — This operation will obtain and list the default *gid, base-directory, shell,* and *skeleton-directory* that will be used when adding a new login with the add operation. Any user may execute this operation.

set — This operation will set the default *gid, base-directory, shell,* and *skeleton-directory* to be used when adding a new login with the add operation. Only the superuser may execute this operation.

Options

-y — Perform the requested operation on the global NIS database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

`-p` Execute the `passwd`(1) program to set a password for the user. The `passwd` program will be executed only if all other additions or modifications have succeeded. Using the `-p` option will cause the operation to become interactive.

`-u` *uid* Specifies the UID of the new user. It must be a non-negative decimal integer less than or equal to `MAXUID` , as defined in `<sys/param.h>`. It defaults an available UID greater than 99.

`-g` *gid* Specifies an existing group's integer ID, or character string name. It defines the new user's group membership. It defaults to the group *general* unless the default has been changed with the `set` operation. For the `set` operation, the *gid* specified will become the new default group.

`-d` *home-directory*
Specifies the home directory of the new user. The default home directory is *base-directory/login*.

`-m` Specifies that the system is to attempt to make the *home-directory* for *login*. The contents of the *skeleton-directory* are copied to the newly created *home-directory*. The new directory has ownership, *uid*, and group, *gid*, read/write/execute permissions.

`-c` *comment*
Specifies any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. It is limited to 128 printable characters and should not include colon (:) or newline.

`-s` *shell* Specifies the full pathname of the program that will be used as the user's shell on login. This field is limited to `PATH_MAX` characters, as defined in `<sys/param.h>`. It defaults to `/sbin/sh`. The *shell* must be a valid executable file.

`-l` *new-login*
Specifies a string of printable characters that is the new login name for the user. It may not contain a colon (:) or a newline and must be a unique login name.

`-r` Tells the `delete` operation to remove the user's home directory from the system, if possible. The files and directories under the home directory will no longer be accessible following successful execution of the command.

`-a` Specifies that the `list` operation will print all information about *login* from the password database in an unformatted report.

`-b` *base-directory*
Specifies the directory that is to be used as the base directory in forming *login*'s default *home-directory*. The default *base-directory* is /home.

`-k` *skeleton-directory*
Specifies a directory that contains skeleton information (e.g. a `.profile` file) to copy into the new user's home directory. The directory must exist. The system provides a *skeleton-directory*, /etc/skel, that can be used for this purpose.

`-q` Quiet. Do not print headers for login reports.

`-v` Verbose. Print full login information, including headers. This option is enabled by default.

EXAMPLES

In the following examples, the system administrator performs the following actions: set the default group to *cad* and default base directory to */home/cadusers*; add a user, *vanilla*, which will create the home directory, /home/cadusers/vanilla; modify the passwd information about the user *vanilla*, changing the user id to *999* and the comment to *Vanilla CAD Account*.

```
admuser -o set -g cad -b /home/cadusers
admuser -o add -m vanilla
admuser -o modify -u 999 -c "Vanilla CAD Account" vanilla
```

DIAGNOSTICS

Errors

- The file `/etc/default/defadduser` does not exist and cannot be created.
- The `add`, `delete`, or `modify` operation was unable to update the password database.
- The `set` operation was unable to update the user defaults.

Exit Codes

This section lists the possible exit codes and what they mean.

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

FILES

/etc/passwd Local password database file.

/etc/default/defadduser
Contains the defaults used by the `add` operation.

/etc/skel Default skeleton directory.

NOTES

A diskless workstation that updates his local password file must have `lockd`(1M), the network lock daemon, running. The server must also have the lock daemon running.

DG/UX ships /etc/passwd with permissions 0444 by default. The `admuser` command attempts to ensure that these permissions remain 0444. If a superuser wishes to modify the password database directly, he or she should use `vipw`(1M).

SEE ALSO

`csh`(1), `login`(1), `sh`(1), `passwd`(1), `yppasswd`(1), `lockd`(1M), `sysadm`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M), `vipw`(1M), `passwd`(4), `profile`(4).

NAME

admxterminal - manage serving of X display terminals

SYNOPSIS

admxterminal -o add [-b *bootstrap*] *hostname* ...

admxterminal -o modify [-b *bootstrap*] *hostname* ...

admxterminal -o delete *hostname* ...

admxterminal -o list [-qv] [*hostname* ...]

DESCRIPTION

All X display terminals, when they begin operation, must obtain a bootstrap from a bootstrap file on some computer system, called the "server". The server machine must recognize requests from X terminal clients, and provide the appropriate bootstrap. The admxterminal command is run on the server machine to identify X display terminals that the machine must serve and the bootstrap file to use for each. The command can also be used to delete an X display terminal from the list of terminals that are to be served, and it can list all X terminals that are being served.

Operations

add — Identify an X terminal, using its network host name, to be served by this machine. A bootstrap file may also be specified. The X terminal must be registered in both the hosts database (using admhost(1M)) and the ethers database (using admether(1M)) before being added as a client.

modify — Change the bootstrap file used for an X terminal.

delete — Remove an X terminal, given its network host name, from the list of terminals that must be served by this machine.

list — Display information about all or selected X terminals. The information includes the network address and the bootstrap file.

Options

-b *bootstrap*
: The pathname of the bootstrap file. The default bootstrap file is /usr/opt/X11/xtd/avx30boot.

-q — Quiet. Omit header lines from the output of the list operation.

-v — Verbose. Include header lines in the output of the list operation. This option is enabled by default.

EXAMPLES

To add an X terminal named "xterm1" using the default bootstrap file:

```
admxterminal -o add xterm1
```

To list all X terminals served by this host:

```
admxterminal -o list
```

OUTPUT

The output of the list operation includes each client host name, client Internet address, name of bootstrap link in the /tftpboot directory (the hexadecimal representation of the Internet address), and the actual bootstrap file.

Unless the -q option is used, a message is displayed if there are no X display terminals being served and header lines are produced if there are.

FILES

`/usr/opt/X11/xtd/avx30boot`	The default bootstrap file.
`/tftpboot`	Directory containing bootstrap file links for each client.

DIAGNOSTICS

Warnings

- An attempt was made to add an X terminal that is already being served by this machine.
- The bootstrap file for the X terminal cannot be removed during a `delete` operation, but the remainder of the operation completed successfully.

Errors

- The X terminal is not registered in the hosts or ethers database.
- The X terminal is not a client of this machine, or it is a diskless client and not an X terminal client (`delete`).
- The bootstrap file does not exist.
- The bootstrap file link could not be created.

Exit Codes

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions. The `add`, `modify`, and `delete` operations can only be used by the super-user.

3 There was an error in the command line.

SEE ALSO

`admether`(1M), `admhost`(1M), `sysadm`(1M), `X`(1), `xtdstart`(1).

NAME

arp – address resolution display and control

SYNOPSIS

```
/usr/bin/arp [ -i dev ] -a
/usr/bin/arp [ -i dev ] host
/usr/bin/arp [ -i dev ] -d host
/usr/bin/arp [ -i dev ] -s host hardware_addr [ temp ] [ pub ]
/usr/bin/arp [ -i dev ] -f file
```

DESCRIPTION

The arp program displays and modifies the Internet-to-hardware address translation tables used by the Address Resolution Protocol arp(6P) and the Reverse Address Resolution Protocol rarp(6P). The hardware address is the Ethernet, 802.3, or token ring address.

When you use the -a option, the program displays all of the current ARP entries by reading the internal kernel tables (using the appropriate ioctl calls). When you use this option, you do not have to specify the name of a *host* or of a *file*.

With *host* as the argument, the program displays the current ARP entry for that host. You may specify the host by name or by number, using Internet standard dot notation.

With the -d option, a superuser may delete an entry for the host named *host*.

Use the -s option to create an ARP entry for the host named *host* with the hardware address *hardware_addr*. The hardware address is specified as six hexadecimal bytes separated by colons. The resulting ARP entry is permanent unless the word temp is specified on the command line. If the word pub is specified, the entry will be "published"; that is, this system will act as an ARP server, responding to requests for *host* even though the host address is not its own. Only a superuser may set a new entry in the table.

The -f option causes the file named *file* to be read and multiple entries to be set in the ARP tables. Only the superuser may use this option. Entries in the file should be of the following form:

host hardware_addr [temp] [pub]

with argument meanings as described above.

If you specify the -i option, only the ARP table for the interface named *dev* will be searched.

SEE ALSO

ifconfig(1M), inet(6F), arp(6P), rarp(6P).

NAME

automount – automatically mount NFS file systems

SYNOPSIS

automount [-mnTv] [-D *name=value*] [-f *master-file*]
[-M *mount-directory*] [-tl *duration*] [-tm *interval*] [-tw *interval*]
[*directory map* [*–mount-options*]] ...

DESCRIPTION

automount is a daemon that automatically and transparently mounts an NFS file system as needed. It monitors attempts to access directories that are associated with an automount map, along with any directories or files that reside under them. When a file is to be accessed, the daemon mounts the appropriate NFS file system. You can assign a map to a directory using an entry in a direct automount map, or by specifying an indirect map on the command line.

The automount daemon appears to be an NFS server to the kernel. automount uses the map to locate an appropriate NFS file server, exported file system, and mount options. It then mounts the file system in a temporary location, and creates a symbolic link to the temporary location. If the file system is not accessed within an appropriate interval (five minutes by default), the daemon unmounts the file system and removes the symbolic link. If the indicated directory has not already been created, the daemon creates it, and then removes it upon exiting.

Since the name-to-location binding is dynamic, updates to an automount map are transparent to the user. This obviates the need to "pre-mount" shared file systems for applications that have "hard coded" references to files.

If the *directory* argument is a pathname, the *map* argument must be an *indirect* map. In an indirect map the key for each entry is a simple name that represents a symbolic link within *directory* to an NFS mount point.

If the *directory* argument is '/–', the map that follows must be a *direct* map. A direct map is not associated with a single directory. Instead, the key for each entry is a full pathname that will itself appear to be a symbolic link to an NFS mount point.

A map can be a file or a Network Interface Service (NIS) map; if a file, the *map* argument must be a full pathname.

The *–mount-options* argument, when supplied, is a comma-separated list of mount(1M) options, preceded by a '–'. If these options are supplied, they become the default mount options for all entries in the map. Mount options provided within a map entry override these defaults.

OPTIONS

-m Suppress initialization of *directory-map* pairs listed in the auto.master NIS database.

-n Disable dynamic mounts. With this option, references through the automount daemon only succeed when the target filesystem has been previously mounted. This can be used to prevent NFS servers from cross-mounting each other.

-T Trace. Expand each NFS call and display it on the standard output.

-v Verbose. Log status and/or warning messages to the console.

-D *envar=value*
Assign *value* to the indicated automount (environment) variable.

`-f` *master-file*
Read a local file for initialization, ahead of the `auto.master` NIS map.

`-M` *mount-directory*
Mount temporary file systems in the named directory, instead of `/tmp_mnt`.

`-tl` *duration*
Specify a *duration*, in seconds, that a file system is to remain mounted when not in use. The default is 5 minutes.

`-tm` *interval*
Specify an *interval*, in seconds, between attempts to mount a filesystem. The default is 30 seconds.

`-tw` *interval*
Specify an *interval*, in seconds, between attempts to unmount filesystems that have exceeded their cached times. The default is 1 minute.

ENVIRONMENT

Environment variables can be used within an `automount` map. For instance, if $HOME appeared within a map, `automount` would expand it to its current value for the HOME variable. Environment variables are expanded only for the automounter's environment — not for the environment of a user using the automounter's services.

The special reference to $ARCH expands to the output of `arch`(1). This can be useful in creating a map entry for mounting executables using a server's export pathname that varies according to the architecture of the client reading the map.

If a reference needs to be protected from affixed characters, you can surround the variable name with curly braces.

USAGE

Map Entry Format

A simple map entry (mapping) takes the form:

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

where *key* is the full pathname of the directory to mount when used in a direct map, or simple name in an indirect map. *mount-options* is a comma-separated list of `mount` options, and *location* specifies a remote filesystem from which the directory may be mounted. In the simple case, *location* takes the form:

hostname : *pathname*

Replicated Filesystems

Multiple *location* fields can be specified for replicated read-only filesystems, in which case `automount` sends multiple `mount` requests; `automount` mounts the file system from the first host that replies to the `mount` request. This request is first made to the local net or subnet. If there is no response, any connected server may respond. Since `automount` does not monitor the status of the server while the filesystem is mounted it will not use another location in the list if the currently mounted server crashes. This support for replicated filesystems is available only at mount time.

If each *location* in the list shares the same *pathname* then a single *location* may be used with a comma-separated list of hostnames.

hostname,*hostname* ... : *pathname*

Sharing Mounts

If *location* is specified in the form:

hostname:*pathname*:*subdir*

hostname is the name of the server from which to mount the file system, *pathname* is the pathname of the directory to mount, and *subdir*, when supplied, is the name of a subdirectory to which the symbolic link is made. This can be used to prevent duplicate mounts when multiple directories in the same remote file system may be accessed. With a map for `/home` such as:

```
able    homeboy:/home/homeboy:able
baker   homeboy:/home/homeboy:baker
```

and a user attempting to access a file in `/home/able`, `automount` mounts `homeboy:/home/homeboy`, but creates a symbolic link called `/home/able` to the `able` subdirectory in the temporarily-mounted filesystem. If a user immediately tries to access a file in `/home/baker`, `automount` needs only to create a symbolic link that points to the `baker` subdirectory; `/home/homeboy` is already mounted.

With the following map:

```
able    homeboy:/home/homeboy/able
baker   homeboy:/home/homeboy/baker
```

`automount` would have to mount the filesystem twice.

Comments and Quoting

A mapping can be continued across input lines by escaping the NEWLINE with a backslash. Comments begin with a `#` and end at the subsequent NEWLINE.

Characters that have special significance to the `automount` map parser may be protected either with double quotes (") or by escaping with a backslash (\). Pathnames with embedded whitespace, colons (:) or dollar ($) should be protected.

Directory Pattern Matching

The '`&`' character is expanded to the value of the *key* field for the entry in which it occurs. In this case:

```
able    homeboy:/home/homeboy:&
```

the `&` expands to `able`.

The '`*`' character, when supplied as the *key* field, is recognized as the catch-all entry. Such an entry will be used if any previous entry has not successfully matched the key being searched for. For instance, if the following entry appeared in the indirect map for `/home`:

```
*       &:/home/&
```

this would allow automatic mounts in `/home` of any remote file system whose location could be specified as:

hostname:`/home/`*hostname*

Multiple Mounts

A multiple mount entry takes the form:

key [/[*mountpoint* [*-mount-options*] *location* ...] ...

The initial / within the '/[*mountpoint*]' is required; the optional *mountpoint* is taken as a pathname relative to the destination of the symbolic link for *key*. If *mountpoint* is omitted in the first occurrence, a *mountpoint* of / is implied.

Given the direct map entry:

```
/arch/src  \
  /         -ro,intr   arch:/arch/src          alt:/arch/src  \
```

```
/1.0      -ro,intr   alt:/arch/src/1.0       arch:/arch/src/1.0  \
/1.0/man  -ro,intr   arch:/arch/src/1.0/man  alt:/arch/src/1.0/man
```

automount would automatically mount /arch/src, /arch/src/1.0 and /arch/src/1.0/man, as needed, from either arch or alt, whichever host responded first. If the mounts are hierarchically related mounts closer to the root must appear before submounts. All the mounts of a multiple mount entry will occur together and will be unmounted together. This is important if the filesystems reference each other with relative symbolic links. Multiple mount entries can be used both in direct maps and in indirect maps.

Included Maps

The contents of another map can be included within a map with an entry of the form:

+mapname

mapname can either be a filename, or the name of an NIS map, or one of the special maps described below. If the key being searched for is not located in an included map, the search continues with the next entry.

Special Maps

There are two special maps currently available: -hosts, and -null. The -hosts map uses the NIS hosts.byname map to locate a remote host when the hostname is specified. This map specifies mounts of all exported file systems from any host. For instance, if the following automount command is already in effect:

```
automount  /net  -hosts
```

then a reference to /net/hermes/usr would initiate an automatic mount of all file systems from hermes that automount can mount; references to a directory under /net/hermes will refer to the corresponding directory relative to hermes root.

The -null map, when indicated on the command line, cancels any subsequent map for the directory indicated. It can be used to cancel a map given in auto.master or for a mount point specified as an entry in a direct map.

Configuration and the auto.master Map

automount normally consults the auto.master NIS configuration map for a list of initial automount maps, and sets up automatic mounts for them in addition to those given on the command line. If there are duplications, the command-line arguments take precedence over a local -f master map and they both take precedence over an NIS auto.master map. This configuration database contains arguments to the automount command, rather than mappings; unless -f is in effect, automount does *not* look for an auto.master file on the local host.

Maps given on the command line, or those given in a local auto.master file specified with -f override those in the NIS auto.master map. For instance, given the command:

```
automount -f  /etc/auto.master /home -null   /-  /etc/auto.direct
```

and a file named /etc/auto.master that contains:

```
/home    auto.home
```

automount would ignore /home entry in /etc/auto.master.

FILES

`/tmp_mnt` directory under which filesystems are dynamically mounted

SEE ALSO

`df`(1M), `ls`(1), `mount`(1M), `stat`(2), `passwd`(4).

NOTES

The `-hosts` map must mount all the exported filesystems from a server. If frequent access to just a single filesystem is required it is more efficient to access the filesystem with a map entry that is tailored to mount just the filesystem of interest.

When it receives signal number 1, `SIGHUP`, `automount` rereads the `/etc/mtab` file to update its internal record of currently-mounted file systems. If a file system mounted with `automount` is unmounted by a `umount` command, `automount` should be forced to reread the file.

An `ls`(1) listing of the entries in the directory for an indirect map shows only the symbolic links for currently mounted filesystems. This restriction is intended to avoid unnecessary mounts as a side effect of programs that read the directory and `stat`(2) each of the names.

Mount points for a single automounter must not be hierarchically related. `automount` will not allow an automount mount point to be created within an automounted filesystem.

`automount` must not be terminated with the $_{\text{SIGKILL}}$ signal (`kill -9`). Without an opportunity to unmount itself, the `automount` mount points will appear to the kernel to belong to a non-responding NFS server. The recommended way to terminate `automount` services is to send a $_{\text{SIGTERM}}$ (`kill -15`) signal to the daemon. This allows the automounter to catch the signal and unmount not only its daemon but also any mounts in `/tmp_mnt`. Mounts in `/tmp_mnt` that are busy will not be unmounted.

Since each direct map entry results in a separate mount for the mount daemon such maps should be kept short. Entries added to a direct map will have no effect until the automounter is restarted.

Entries in both direct and indirect maps can be modified at any time. The new information will be used when `automount` next uses the map entry to do a mount. `automount` does not cache map entries.

The Network Information Service (NIS) was formerly known as Sun Yellow Pages (YP). The functionality of the two remains the same; only the name has changed.

BUGS

The `bg` mount option is not recognized by the automounter.

Since `automount` is single-threaded, any request that is delayed by a slow or non-responding NFS server will delay all subsequent automatic mount requests until it completes.

Programs that read `/etc/mtab` and then touch files that reside under automatic mount points will introduce further entries to the file.

Automatically-mounted file systems are mounted with type `ignore`; they do not appear in the output of either `mount`(1M), or `df`(1).

NAME

autopush - configure automatically pushed STREAMS modules

SYNOPSIS

autopush -f *file*

autopush -r -M *major* -m *minor*

autopush -g -M *major* -m *minor*

where:

file An autopush configuration script, either the path to a file or - to indicate standard input

major A STREAMS major device identifier, either a major device number or the path to a device file

minor A STREAMS minor device identifier, either a minor device number or the path to a device file

DESCRIPTION

The autopush command configures the list of modules to be automatically pushed onto a stream when a STREAMS device is opened. It can also be used to remove a previous setting or get information on a setting.

STREAMS devices are specified to autopush using "device identifiers" that represent major and minor device numbers. A device identifier can be an explicit non-negative number. Alternatively, it can be the path of a character-special device file, in which case the major or minor device number of the node is used (whichever is appropriate to the context). The latter choice is usually more convenient and less prone to change (see the **NOTES** section below).

Options are:

-f *file* Set up the autopush configuration for each driver according to the information stored in the file specified by *file*. If - is given for *file*, autopush reads the information from its standard input. The information consists of lines of at least four fields each, where the fields are separated by blanks (spaces or tabs) as shown below:

major minor last_minor mod1 mod2 ... modN

The first three fields are device identifiers. *major* specifies the major device number, *minor* specifies the (first or only) minor device number, and *last_minor* specifies the last minor device number.

If *minor* is -1 (generic), then all minor devices of a driver specified by *major* are configured and the value for *last_minor* is ignored. If *last_minor* is 0 or -1, then only the single minor device specified by *minor* is configured. If *minor* is less than *last_minor*, all minor devices in the range *minor* through *last_minor* (inclusive) are configured for *major*,

The remaining fields are strings representing the names of STREAMS modules. Each name is separated by a space. A maximum of eight modules can be automatically pushed onto a stream. The modules are pushed in the order they are specified (leftmost first).

Comment lines starting with a # sign are ignored by autopush. Blank lines can also be included in the file.

-r This option removes the previous configuration setting associated with the particular *major* and *minor* device identifiers specified by the -M and -m

options, respectively. If the values of *major* and *minor* correspond to the setting of a range of minor devices, where *minor* matches the first minor device number in the range, the configuration would be removed for the entire range.

`-g` This option gets (prints) the current configuration setting associated with the particular *major* and *minor* device identifiers specified by the `-M` and `-m` options, respectively. It will also print the starting minor device number if the request corresponds to the setting of a range (as described for the `-f` option).

EXAMPLES

```
$ ls -l /dev/tty00
crw--w--w-   1 root   root   19,  0 Feb 21 12:51 /dev/tty00

$ autopush -g -M 19 -m 0
     Major      Minor  Lastminor        Modules
        19          0      -            ldterm ttcompat
```

This example prints out the list of automatically-pushed STREAMS modules for the TTY device `/dev/tty00`. `ls -l` provides the major and minor device numbers (`19` and `0`, respectively) for the device. These numbers can then be used as the arguments to the `-M` and `-m` options of `autopush`. Note that the first minor number is 0 and the last minor number is null. These values indicate that the `autopush` setting holds for only the individual minor device represented by `/dev/tty00` (although other devices may have duplicate settings).

```
autopush -r -M /dev/console -m 0
echo "/dev/console -1 0 ldterm" | autopush -f -
```

This example reconfigures the console device to automatically push one STREAMS module, `ldterm`. The first command line removes the current configuration setting associated with the `console` driver. The second command line creates the new configuration setting, making it apply to all minor devices associated with the `console` driver.

SEE ALSO

`ls(1)`, `strchg(1)`, `sad(7)`, `streamio(7)`

NOTES

On the DG/UX System major and minor device numbers change as devices are configured into and de-configured from the system. Reconfiguration can obsolete explicit device numbers used with `autopush`.

The following features are Data General enhancements: the alternative of using device pathnames instead of major and minor device numbers; the ability to read from standard input instead of from a file; and the option of using `-1` instead of `0` for the *last_minor* field of the configuration file.

NAME

bcs_cat - type hosts, networks, passwd, protocols, group or services information

SYNOPSIS

/etc/bcs_cat *file*

where:

file group, hosts, networks, passwd, protocols, or services

DESCRIPTION

The bcs_cat command types to standard output all of the information that you would normally find in the local /etc/*file* file where *file* is the file specified on the command line. If some of your system's entries for these files reside in /etc/*file* files on another system (such as a network database server), the bcs_cat command will return these remotely-stored entries in addition to those in your local file.

STANDARDS

This command provides functionality required for Binary Compatibility Standard (BCS) compliance.

EXAMPLES

The following command line prints local passwd entries in addition to any stored on your network database server:

```
bcs_cat passwd
```

SEE ALSO

group(4), hosts(4), networks(4), passwd(4), protocols(4), services(4).
88open Binary Compatibility Standard.

NAME

`biod` – start block I/O servers

SYNOPSIS

`/usr/sbin/biod` [*nservers*]

where:

nservers An integer specifying how many servers to start

DESCRIPTION

`Biod` starts *nservers* asynchronous block I/O servers (daemons). The `biod` servers are used in performing asynchronous I/O between secondary storage and main memory except for paging to local swap areas. For example, file readahead, most file buffer writebacks, and diskless paging all use biod servers. The more I/O expected on the system, particularly NFS I/O, the more servers are needed to service it.

A good value for *nservers* for a typical system using NFS is 8; a smaller value may yield equally good performance if the system is not used as an NFS client.

SEE ALSO

`nfs`(4P).

NAME

bootparamd – server for boot parameters

SYNOPSIS

```
/usr/etc/rpc.bootparamd [ -d ]
```

DESCRIPTION

bootparamd is a server process that provides information to diskless clients necessary for booting. It consults the bootparams database. If the client is not found there, or if the Network Information Service (NIS) is not running, then the /etc/bootparams file is consulted.

bootparamd can be invoked either by inetd(1M) or by the user.

OPTIONS

-d Display the debugging information.

FILES

/etc/bootparams

SEE ALSO

inetd(1M), bootparams(4).

NAME

captoinfo – convert a TERMCAP entry into a TERMINFO entry

SYNOPSIS

captoinfo [-v ...] [-V] [-1] [-w *width*] [*file*] ...

where:

width is the maximum output width desired, an integer
file is the pathname of a file in termcap(5) format

DESCRIPTION

The captoinfo utility looks in *file* for termcap(5) descriptions. For each one found, an equivalent terminfo(4) description is written to standard output, along with any comments found. A description which is expressed as relative to another description (as specified by the termcap(5) tc=*term* field) will be reduced to the minimum superset before being output.

If no *file* is given, then the environment variable TERMCAP is used for the filename or entry. If the value of TERMCAP is a full pathname to a file, only the terminal named by the value of the environment variable TERM is extracted from that file. Otherwise, the value of TERMCAP is interpreted as one or more termcap(5) entries, all of which are extracted. If the environment variable TERMCAP is not set, then the default termcap(5) file /usr/share/lib/termcap is read.

Options are:

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

-V Print out the version of the program in use on standard error, and exit.

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

-w *width*
Change the maximum output width to *width* characters.

FILES

/usr/share/lib/terminfo/?/*
compiled terminal description database

/usr/share/lib/termcap
old, textual terminal description database

DIAGNOSTICS

Command line syntax errors cause captoinfo to print a usage message to standard error. Other errors cause captoinfo to produce diagnostic messages as described below. Most of these messages are preceded by the name of the entry causing the error.

tgetent failed with return code *n* (*reason*).

The termcap(5) entry is not valid for the specified *reason*. In particular, check for an invalid tc= field.

commented out code *'cc'* is unknown.

An unknown commented out termcap code *cc* was encountered.

unknown type given for the termcap code *'cc'*.

The termcap(5) description had a field *cc* for which the type was not boolean, numeric, or string.

`wrong type given for the boolean termcap code 'cc'.`

`wrong type given for the numeric termcap code 'cc'.`

`wrong type given for the string termcap code 'cc'.`

The `termcap` field *cc* was entered as a capability of the wrong type; for instance, a boolean field was entered as a numeric capability.

`the boolean termcap code 'cc' is not a valid name.`

`the numeric termcap code 'cc' is not a valid name.`

`the string termcap code 'cc' is not a valid name.`

An unknown `termcap`(5) code *cc* was encountered.

`cap` *cc* `(info` *ii*`) is NULL: REMOVED`

The `termcap`(5) field *cc*, which translates to `terminfo`(4) capability *ii*, was specified as a null string. The correct way to cancel a field is to follow it with the `@` character; for example, `:bs@:`. Giving a null string could cause incorrect assumptions to be made by software which uses `termcap`(5) or `terminfo`(4).

`obsolete 2 character name 'ss' removed.`
`synonyms are: 'string'`

An obsolete two-letter terminal name *ss* in the current `termcap`(5) entry was not propagated into the `terminfo`(4) entry. The synonyms listed in *string* will still be incorporated, though.

`a function key for 'cc' was specified with the value` *vv1*`, but it already has the value` *vv2*`.`

When parsing the `ko` field, the key *cc* was specified as having the same value as the capability *vv1*, but *cc* already had the value *vv2*.

`the unknown termcap name 'cc' was`
`specified in the 'ko' termcap capability.`

A key *cc* which could not be handled was specified in the `ko` capability.

`the vi character 'v' (info 'ii') has the value 'xx', but 'ma' gives 'n'.`

The `ma` field specified a function *v*, equivalent to `terminfo`(4) capability *ii*, with a value *n*. However, the key represented by *v* had previously been set to a different value *xx*.

`the unknown vi key 'v' was`
`specified in the 'ma' termcap capability.`

A key unknown to `captoinfo` was specified in the `ma` capability.

`Warning: termcap sg and ug had different values (`*nn1*`<->`*nn2*`).`

The `termcap`(5) value for the `sg` field (*nn1*) was different from the value for the `ug` field (*nn2*), but `terminfo`(4) can represent only one of these values; `sg` translates into `xmc`, which is assumed to have the same value as `ug`).

`Warning: the string produced for 'ii' may be inefficient.`

The parameterized string created for field *ii* should be rewritten by hand.

`Null term_name given.`

The terminal type was null. This message is printed if the environment variable `TERM` is not set or is null.

`cannot open` *file* `for reading.`

The file *file* could not be opened for read-only access.

`the boolean name '`*info*`' was not found!`

`the numeric name '`*info*`' was not found!`

`the string name '`*info*`' was not found!`

The `terminfo`(4) capability *info* does not exist. This message indicates an internal error in `captoinfo`.

SEE ALSO

`infocmp`(1M), `tic`(1M), `curses`(3X), `terminfo`(4), `termcap`(5)

NOTES

`Captoinfo` should be used to convert `termcap`(5) entries to `terminfo`(4) entries because programs that use `curses`(3X) depend upon `terminfo`(4).

Certain `termcap`(5) defaults are assumed to be true. For example, the bell character (`terminfo`(4) `bel`) is assumed to be `^g`. The linefeed capability (`termcap`(5) `nl`) is assumed to be the same for both `cursor_down` and `scroll_forward` (`terminfo`(4) `cud1` and `ind`, respectively). Padding information is assumed to belong at the end of the string.

The algorithm used to expand parameterized information for `termcap`(5) fields such as `cursor_position` (`termcap`(5) `cm`, `terminfo`(4) `cup`) will sometimes produce a string which, though technically correct, may not be optimal. In particular, the rarely used `termcap`(5) operation `%n` will produce strings that are especially long. Most occurrences of these non-optimal strings will be flagged with a warning message and may need to be recoded by hand.

The short two-letter name at the beginning of the list of names in a `termcap`(5) entry, a hold-over from an earlier version of the `termcap`(5) convention, will be deleted.

NAME

chroot - change root directory for a command

SYNOPSIS

/usr/sbin/chroot *newroot command*

DESCRIPTION

Chroot executes *command* relative to *newroot*. After executing chroot, the initial slash (/) in subsequent pathnames is changed to the new root directory you specify. *newroot* becomes the initial working directory.

The new root is always relative to the current root. If a chroot is currently in effect (for example, a sh or csh command), *newroot* is relative to the current root of the running process, not the original root (/).

Changing the root for *command* does not change the root for chroot. Thus, I/O redirection is relative to the old root directory.

Only the superuser can use the chroot command.

EXAMPLES

```
/usr/sbin/chroot /usr/alex/test /grep pattern /file1 > grep.out

/usr/alex/test/grep pattern /usr/alex/test/file1 > grep.out
```

These two lines are equivalent. Note that I/O redirection is relative to the original root, not the new one.

SEE ALSO

chdir(2), chroot(2).

NOTE

Be careful if you wish to reference special files in the new root file system. Unless the new root is /dev or you have copies of the /dev files within the range of the new root, these special files will be inaccessible.

NAME

chrtbl – generate character classification and conversion tables

SYNOPSIS

chrtbl [*file*]
chrtbl -d [*ctype_file* [*num_file*]]

DESCRIPTION

The chrtbl command can be used two ways: without the -d option, to create tables of character classification information; and with the -d option to dump a text version of such tables.

The chrtbl -d *ctype_file num_file* command dumps to it's standard output a text version of the LC_CTYPE character table in file *ctype_file* and the LC_NUMERIC numeric information table in file *num_file*. If *num_file*, or both *ctype_file* and *num_file* are not specified, the corresponding table from the current locale is dumped. You can modify the resulting text file, and use it as input to chrtbl, to produce modified LC_CTYPE and LC_NUMERIC files. These files may be used to either replace the existing LC_CTYPE and LC_NUMERIC files in an existing locale, or to create a new locale. However, you must never modify any of the files (including LC_CTYPE and LC_NUMERIC) in /usr/lib/locale/C, the C locale.

The chrtbl command without the -d option creates two tables containing information on character classification, upper/lower-case conversion, character-set width, and numeric formatting. One table is an array of (257*2) + 7 bytes that is encoded so a table lookup can be used to determine the character classification of a character, convert a character [see ctype(3C)], and find the byte and screen width of a character in one of the supplementary code sets. The other table contains information about the format of non-monetary numeric quantities: the first byte specifies the decimal delimiter; the second byte specifies the thousands delimiter; and the remaining bytes comprise a null terminated string indicating the grouping (each element of the string is taken as an integer that indicates the number of digits that comprise the current group in a formatted non-monetary numeric quantity).

chrtbl reads the user-defined character classification and conversion information from *file* and creates three output files in the current directory. To construct *file*, use the file supplied in /usr/lib/locale/C/chrtbl_C, or the output of chrtbl -d as a starting point. You may add entries, but do not change the original values supplied with the system. For example, for other locales you may wish to add eight-bit entries to the ASCII definitions provided in this file.

One output file, ctype.c (a C-language source file), contains a (257*2)+7-byte array generated from processing the information from *file*. You should review the content of ctype.c to verify that the array is set up as you had planned. (In addition, an application program could use ctype.c.) The first 257 bytes of the array in ctype.c are used for character classification. The characters used for initializing these bytes of the array represent character classifications that are defined in /usr/include/ctype.h; for example, _L means a character is lower case and _S|_B means the character is both a spacing character and a blank. The second 257 bytes of the array are used for character conversion. These bytes of the array are initialized so that characters for which you do not provide conversion information will be converted to themselves. When you do provide conversion information, the first value of the pair is stored where the second one would be stored normally, and vice versa; for example, if you provide <0x41 0x61>, then 0x61 is stored where 0x41 would be stored normally, and 0x61 is stored where 0x41 would be stored normally. The last 7 bytes are used for character width information for up to three

supplementary code sets.

The second output file (a data file) contains the same information, but is structured for efficient use by the character classification and conversion routines (see ctype(3C)). The name of this output file is the value you assign to the keyword LC_CTYPE read in from *file*. Before this file can be used by the character classification and conversion routines, it must be installed in the /usr/lib/locale/*locale* directory with the name LC_CTYPE by someone who is super-user or a member of group bin. This file must be readable by user, group, and other; no other permissions should be set. To use the character classification and conversion tables in this file, set the LC_CTYPE environment variable appropriately (see environ(5) or setlocale(3C)).

The third output file (a data file) is created only if numeric formatting information is specified in the input file. The name of this output file is the value you assign to the keyword LC_NUMERIC read in from *file*. Before this file can be used, it must be installed in the /usr/lib/locale/*locale* directory with the name LC_NUMERIC by someone who is super-user or a member of group bin. This file must be readable by user, group, and other; no other permissions should be set. To use the numeric formatting information in this file, set the LC_NUMERIC environment variable appropriately (see environ(5) or setlocale(3C)).

The name of the locale where you install the files LC_CTYPE and LC_NUMERIC should correspond to the conventions defined in *file*. For example, if French conventions were defined, and the name for the French locale on your system is french, then you should install the files in /usr/lib/locale/french.

If no input file is given, or if the argument "–" is encountered, chrtbl reads from standard input.

The syntax of *file* allows the user to define the names of the data files created by chrtbl, the assignment of characters to character classifications, the relationship between upper and lower-case letters, byte and screen widths for up to three supplementary code sets, and three items of numeric formatting information: the decimal delimiter, the thousands delimiter and the grouping. The keywords recognized by chrtbl are:

LC_CTYPE	name of the data file created by chrtbl to contain character classification, conversion, and width information
isupper	character codes to be classified as upper-case letters
islower	character codes to be classified as lower-case letters
isdigit	character codes to be classified as numeric
isspace	character codes to be classified as spacing (delimiter) characters
ispunct	character codes to be classified as punctuation characters
iscntrl	character codes to be classified as control characters
isblank	character code for the blank (space) character
isxdigit	character codes to be classified as hexadecimal digits
ul	relationship between upper- and lower-case characters
cswidth	byte and screen width information (by default, each is one character wide)
LC_NUMERIC	name of the data file created by chrtbl to contain numeric formatting information

`decimal_point` decimal delimiter

`thousands_sep` thousands delimiter

`grouping` string in which each element is taken as an integer that indicates the number of digits that comprise the current group in a formatted non-monetary numeric quantity.

Any lines with the number sign (#) in the first column are treated as comments and are ignored. Blank lines are also ignored.

Characters for `isupper`, `islower`, `isdigit`, `isspace`, `ispunct`, `iscntrl`, `isblank`, `isxdigit`, and `ul` can be represented as a hexadecimal or octal constant (for example, the letter `a` can be represented as `0x61` in hexadecimal or `0141` in octal). Hexadecimal and octal constants may be separated by one or more space and/or tab characters.

The dash character (-) may be used to indicate a range of consecutive numbers. Zero or more space characters may be used for separating the dash character from the numbers.

The backslash character (\) is used for line continuation. Only a carriage return is permitted after the backslash character.

The relationship between upper- and lower-case letters (`ul`) is expressed as ordered pairs of octal or hexadecimal constants: *<upper-case_character lower-case_character>*. These two constants may be separated by one or more space characters. Zero or more space characters may be used for separating the angle brackets (< >) from the numbers.

The following is the format of an input specification for `cswidth`:
n1:s1,n2:s2,n3:s3
where,

n1 byte width for supplementary code set 1, required
s1 screen width for supplementary code set 1
n2 byte width for supplementary code set 2
s2 screen width for supplementary code set 2
n3 byte width for supplementary code set 3
s3 screen width for supplementary code set 3

`decimal_point` and `thousands_sep` are specified by a single character that gives the delimiter. `grouping` is specified by a quoted string in which each member may be in octal or hex representation. For example, `\3` or `\x3` could be used to set the value of a member of the string to 3.

EXAMPLE

The following is an example of an input file used to create the USA-ENGLISH code set definition table in a file named `usa` and the non-monetary numeric formatting information in a file name `num-usa`.

```
LC_CTYPE  usa
isupper   0x41 - 0x5a
islower   0x61 - 0x7a
isdigit   0x30 - 0x39
isspace   0x20 0x9 - 0xd
ispunct   0x21 - 0x2f 0x3a - 0x40    \
          0x5b - 0x60 0x7b - 0x7e
iscntrl   0x0 - 0x1f  0x7f
isblank   0x20
isxdigit  0x30 - 0x39 0x61 - 0x66    \
```

```
                0x41 - 0x46
        ul      <0x41 0x61> <0x42 0x62> <0x43 0x63>  \
                <0x44 0x64> <0x45 0x65> <0x46 0x66>  \
                <0x47 0x67> <0x48 0x68> <0x49 0x69>  \
                <0x4a 0x6a> <0x4b 0x6b> <0x4c 0x6c>  \
                <0x4d 0x6d> <0x4e 0x6e> <0x4f 0x6f>  \
                <0x50 0x70> <0x51 0x71> <0x52 0x72>  \
                <0x53 0x73> <0x54 0x74> <0x55 0x75>  \
                <0x56 0x76> <0x57 0x77> <0x58 0x78>  \
                <0x59 0x79> <0x5a 0x7a>
        cswidth         1:1,0:0,0:0
        LC_NUMERIC      num_usa
        decimal_point           .
        thousands_sep           ,
        grouping                        "\3"
```

FILES

/usr/lib/locale/*locale*/LC_CTYPE
: data files containing character classification, conversion, and character-set width information created by `chrtbl`

/usr/lib/locale/*locale*/LC_NUMERIC
: data files containing numeric formatting information created by `chrtbl`

/usr/include/ctype.h
: header file containing information used by character classification and conversion routines

/usr/lib/locale/C/chrtbl_C
: input file used to construct `LC_CTYPE` and `LC_NUMERIC` in the default locale.

DIAGNOSTICS

The error messages produced by `chrtbl` are intended to be self-explanatory. They indicate errors in the command line or syntactic errors encountered within the input file.

SEE ALSO

ctype(3C), setlocale(3C), environ(5).

CAUTION

Changing the files in `/usr/lib/locale/C` will cause the system to behave unpredictably.

NAME

ckbinarsys - determine whether remote system can accept binary messages

SYNOPSIS

ckbinarsys [-S] -s *remote_system_name* -t *content_type*

DESCRIPTION

Because rmail can transport binary data, it may be important to determine whether a particular remote system (typically the next hop) can handle binary data via the chosen transport layer agent (uux, SMTP, etc.)

ckbinarsys consults the file /etc/mail/binarsys for information on a specific remote system. ckbinarsys returns its results via an appropriate exit code. An exit code of zero implies that it is OK to send a message with the indicated content type to the system specified. An exit code other than zero indicates that the remote system cannot properly handle messages with binary content.

The absence of the binarsys file will cause ckbinarsys to exit with a non-zero exit code.

Command-line arguments are:

-s *remote_system_name*
: Name of remote system to look up in /etc/mail/binarsys

-t *content_type*
: Content type of message to be sent. When invoked by rmail, this will be one of two strings: text or binary, as determined by mail independent of any Content-Type: header lines that may be present within the message header. All other arguments are treated as equivalent to binary.

-S
: Normally, ckbinarsys will print a message (if the binary mail is rejected) which would be suitable for rmail to return in the negative acknowledgement mail. When -S is specified, no message will be printed.

FILES

/etc/mail/binarsys
/usr/lib/mail/surrcmd/ckbinarsys

SEE ALSO

mail(1), uux(1) in the *User's Reference*; binarsys(4), mailsurr(4) in the *Programmer's Reference*.

NAME

clri – clear inode

SYNOPSIS

/etc/clri *filesys*|*special* *i-number* ...

where:

filesys The pathname of the directory with which the file system is associated in the file /etc/fstab

special The pathname of a special file referring to a device containing an unmounted file system

i-number An inode number

DESCRIPTION

Clri writes zeros on the bytes occupied by the inode numbered *i-number*. The inode becomes allocatable.

After clri is executed, any blocks in the affected file will show up as unallocated in an fsck(1M) of the file system. Use clri very carefully and only in emergencies, since it can introduce inconsistencies into the file system.

Read and write permission is required on the specified *file-system* device.

The primary purpose of this command is to remove a file that doesn't appear in any directory. If you use it to clear an inode that does appear in a directory, you should also remove the directory entry. Otherwise, when the inode is reallocated to a new file, the old entry will still point to that file. If you remove the old entry then, the new file will be destroyed. Since the new entry will again point to an unallocated inode, the whole cycle is likely to be repeated.

SEE ALSO

fsck(1M), fsdb(1M), ncheck(1M), fs(4).

NOTES

Fsck automatically removes files that don't appear in any directory. Whenever possible, use fsck instead of clri.

Clri cannot be run on the root file system because that file system cannot be unmounted.

NAME

colltbl - create collation database

SYNOPSIS

colltbl [*file* | -]
colltbl -d [*file*]

DESCRIPTION

The colltbl command without the -d option takes as input a specification file, *file*, that describes the collating sequence for a particular language and creates a database that can be read by strxfrm(3C) and strcoll(3C). strxfrm(3C) transforms its first argument and places the result in its second argument. The transformed string is such that it can be correctly ordered with other transformed strings by using strcmp(3C), strncmp(3C) or memcmp(3C). strcoll(3C) transforms its arguments and does a comparison.

If no input file is supplied, *stdin* is read.

The output file produced contains the database with collating sequence information in a form usable by system commands and routines. The name of this output file is the value you assign to the keyword codeset read in from *file*. Before this file can be used, it must be installed in the /usr/lib/locale/*locale* directory with the name LC_COLLATE by someone who is super-user or a member of group bin. *locale* corresponds to the language area whose collation sequence is described in *file*. This file must be readable by user, group, and other; no other permissions should be set. To use the collating sequence information in this file, set the LC_COLLATE or LANG environment variable appropriately (see environ(5) or setlocale(3C)).

With the -d option, colltbl dumps to its standard output a text version of the LC_COLLATE collation table in file *file*. If no input file is specified, the collation table in use for the current locale is dumped. You can modify the resulting text file, and use it as input to colltbl, to produce a modified LC_COLLATE collation table file. This file may be used to either replace the existing LC_COLLATE file in an existing locale, or to create a new locale. However, you must never modify any of the files (including LC_COLLATE) in /usr/lib/locale/C, the C locale.

The colltbl command can support languages whose collating sequence can be completely described by the following cases:

- Ordering of single characters within the codeset. For example, in Swedish, V is sorted after U, before X and with W (V and W are considered identical as far as sorting is concerned).
- Ordering of "double characters" in the collation sequence. For example, in Spanish, ch and ll are collated after c and l, respectively.
- Ordering of a single character as if it consists of two characters. For example, in German, the "sharp s", β, is sorted as ss. This is a special instance of the next case below.
- Substitution of one character string with another character string. In the example above, the string β is replaced with ss during sorting.
- Ignoring certain characters in the codeset during collation. For example, if - were ignored during collation, then the strings re-locate and relocate would be equal.
- Secondary ordering between characters. In the case where two characters are sorted together in the collation sequence, (i.e., they have the same "primary" ordering), there is sometimes a secondary ordering that is used if two strings are

identical except for characters that have the same primary ordering. For example, in French, the letters `e` and `è` have the same primary ordering but `e` comes before `è` in the secondary ordering. Thus the word `lever` would be ordered before `lèver`, but `lèver` would be sorted before `levitate`. (Note that if `e` came before `è` in the primary ordering, then `lèver` would be sorted after `levitate`.)

The specification file consists of three types of statements:

1. `codeset` *filename*

   filename is the name of the output file to be created by `colltbl`.

2. `order is` *order_list*

   order_list is a list of symbols, separated by semicolons, that defines the collating sequence. The special symbol, `...`, specifies symbols that are lexically sequential in a short-hand form. For example,

   ```
   order is      a;b;c;d;...;x;y;z
   ```

   would specify the list of lower_case letters. Of course, this could be further compressed to just `a;...;z`.

   A symbol can be up to two bytes in length and can be represented in any one of the following ways:

   - the symbol itself (e.g., `a` for the lower-case letter `a`),
   - in octal representation (e.g., `\141` or `0141` for the letter `a`), or
   - in hexadecimal representation (e.g., `\x61` or `0x61` for the letter `a`).

   Any combination of these may be used as well.

   The backslash character, `\` , is used for continuation. No characters are permitted after the backslash character.

   Symbols enclosed in parenthesis are assigned the same primary ordering but different secondary ordering. Symbols enclosed in curly brackets are assigned only the same primary ordering. For example,

   ```
   order is     a;b;c;ch;d;(e;è);f;...;z;\
                {1;...;9};A;...;Z
   ```

   In the above example, `e` and `è` are assigned the same primary ordering and different secondary ordering, digits 1 through 9 are assigned the same primary ordering and no secondary ordering. Only primary ordering is assigned to the remaining symbols. Notice how double letters can be specified in the collating sequence (letter `ch` comes between `c` and `d`).

   If a character is not included in the `order is` statement it is excluded from the ordering and will be ignored during sorting.

3. `substitute` *string* `with` *repl*

   The `substitute` statement substitutes the string *string* with the string *repl*. This can be used, for example, to provide rules to sort the abbreviated month names numerically:

```
substitute "Jan" with "01"
substitute "Feb" with "02"
      .
      .
      .
substitute "Dec" with "12"
```

A simpler use of the `substitute` statement that was mentioned above was to substitute a single character with two characters, as with the substitution of β with `ss` in German.

The `substitute` statement is optional. The `order is` and `codeset` statements must appear in the specification file.

Any lines in the specification file with a `#` in the first column are treated as comments and are ignored. Empty lines are also ignored.

EXAMPLE

The following example shows the collation specification required to support a hypothetical telephone book sorting sequence.

The sorting sequence is defined by the following rules:

a. Upper and lower case letters must be sorted together, but upper case letters have precedence over lower case letters.

b. All special characters and punctuation should be ignored.

c. Digits must be sorted as their alphabetic counterparts (e.g., `0` as `zero`, `1` as one).

d. The `Ch`, `ch`, `CH` combinations must be collated between `C` and `D`.

e. `V` and `W`, `v` and `w` must be collated together.

The input specification file to `colltbl` will contain:

```
codesettelephone

order is      A;a;B;b;C;c;CH;Ch;ch;D;d;E;e;F;f;\
              G;g;H;h:I;i;J;j;K;k;L;l;M;m;N;n;O;o;P;p;\
              Q;q;R;r;S;s;T;t;U;u;{V;W};{v;w};X;x;Y;y;Z;z

substitute "0" with "zero"
substitute "1" with "one"
substitute "2" with "two"
substitute "3" with "three"
substitute "4" with "four"
substitute "5" with "five"
substitute "6" with "six"
substitute "7" with "seven"
substitute "8" with "eight"
substitute "9" with "nine"
```

FILES

`/lib/locale/`*`locale`*`/LC_COLLATE`
`LC_COLLATE` database for *locale*

`/usr/lib/locale/C/colltbl_C`
input file used to construct `LC_COLLATE` in the default locale.

SEE ALSO

`memory`(3C), `setlocale`(3C), `strcoll`(3C), `string`(3C), `strxfrm`(3C), `regexpr`(3C), `environ`(5) in the *Programmer's Reference Manual*.

NAME

config - configure a system

SYNOPSIS

/usr/sbin/config [-q] [-c *conf_file*] [-m *master_dir*] *system_file*

where:

conf_file The C language source file to be created with the configuration tables describing the configurable kernel components to be used in the system. The default is conf.c.

master_dir The directory with the master files to be used in configuring the system. All files in this directory will be processed as master files (see master(4)). The default is /usr/etc/master.d.

system_file The device description file (see system(4)). This file describes all the configurable kernel components desired in the kernel to be built.

DESCRIPTION

This program generates a C language source file based on the files (see master(4) and system(4)) which describe the hardware devices, software drivers, STREAMS modules, socket protocols, and tunable parameters on the system. This C source file can then be compiled and linked against kernel libraries to produce a new kernel. Most users will simply provide config with the name of the system file describing their system (see below). Users with greater kernel experience may vary the configuration process with the -c and -m arguments.

Options are:

-q Do not complain about errors.

-c Specify a non-default name for the configuration file; conf.c is the default.

-m Specify a non-default master file directory; /usr/etc/master.d is the default.

FILES

/usr/etc/master.d	default master file directory
conf.c	default output configuration table file

DIAGNOSTICS

Diagnostics are routed to the standard output and are self-explanatory.

SEE ALSO

sysdef(1M), master(4), system(4).
Installing the DG/UX System, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NAME

`crash` – examine system images

SYNOPSIS

`crash` [`-p`] [`-l` *log_file*] [*image_file*] [*sym_tab_file*]

where:

log_file The pathname of a file for logging `crash` input and output.

image_file The name of a file to used as a memory image. This file can be a system image, memory dump, or `/dev/mem`. You must have read permissions for the file (so for `/dev/mem`, you must be superuser). If the file is an executable (system file), the symbol table may be taken from this file.

sym_tab_file The name of a symbol table file, a file containing the executable system image used to produce the memory image file; `/dgux` is the default.

DESCRIPTION

The `crash` interactive utility (located in `/usr/bin`) allows analysis of a system image, dump, or a running system. It can display system databases, look at logical memory, and perform miscellaneous functions that are useful for inspecting a memory dump. Options are:

`-p` Read all data directly from the image file; do not use the symbol table file to read kernel code and read-only data.

`-l` Log all input and output to the specified log file.

Command Summary

Once `crash` has started, you enter a `crash` command line interpreter. `Crash` has four sets of commands: memory/symbol, general, debugger, and support. Following is a summary of these commands by category; alternate short command names are listed where they exist:

Memory and Symbol Commands

These commands let you read a program's symbol table or display its memory. In addition, there are several expression evaluation commands that let you compute the values of octal or hexadecimal expressions. The memory and symbol commands are:

`memread (mr)`
: Read and display memory

`memwrite (mw)`
: View or modify memory

`memsearch (ms)`
: Search memory

`regsearch (rs)`
: Regular expression search

`patdump (pd)`
: Pattern dump

`view (vi)`
: View memory in code format

`down (do)`
: View down

`up`
: View up

`translate (ts)`
: Translate an expression value to a symbol

`name (nm)`
Print symbol table entry

General Commands

The general commands are:

`eval (ev)`
Evaluate expressions
`print` Print values – unformated
`printf` Print values – formatted
`mode` Change the radix of numeric output
`global` Create a 32-bit variable to be used in an expression
`set` Set a global variable to the value of an expression
`help` Print help information (?)

Debugger Commands

These commands report and control the state of the hardware and the kernel execution. (Command abbreviations appear in parentheses.)

`brk` Set breakpoints (`b, bp, br`)
`delete` Delete breakpoints (`d`)
`proceed` Proceed with execution of the kernel (`p`)
`trace` Trace back through a process's stack (`tr`)
`register`
Display the contents of the general registers (`reg`)
`control` Display the contents of the control registers (`ctl`)
`cmmu` Display the contents of the cache and memory management unit
`ltop` Translate a logical to a physical address
`status` Display the status of a physical processor
`vp` Describe a virtual processor
`focus` Look at the address space of a given process (`fo`)
`halt` Halt the system

Support Commands

These commands control the crash program itself.

`quit` Exit from `crash`
`fh` Print program information from the file header
`x` Execute a file containing crash commands

Each command is discussed in this man page. However, in general the syntax of `crash` commands is of the form:

command [*options*] [*arguments*]

Each command must be on one line terminated by a newline, carriage return, formfeed, or null character. The prompt for `crash` is

[*jp*:*program_name*:*vp*]

where:

jp The current job processor number
program_name The name of the currently focused process
vp The virtual processor number of the currently focused process

`Crash` supports a help facility that you invoke with the `help` command.

Terminology

The following terms are used in this document:

System image file	The DG/UX System bootable image, normally called `dgux`. This image file contains the code for the system and the symbol table. A system image is required to execute `crash`.
System dump	A physical memory dump created by running the system shutdown code.
Running system	A system in normal operation. `crash` may be used to analyze this system by reading physical memory from `/dev/mem`.
Address translation	The process of converting a logical address to a physical address.
Slot number	An index by which many structures are referenced into an array. For example, processes are referenced by their index, not their process id, into the process table.
VP	A virtual processor. A VP contains the necessary information to allow a process to run on a processor. There are a limited number of VPs in a system.

Memory and Symbol Commands

These commands allow you to display symbol table data and program memory. All of the memory commands (`mr`, `mw`, `ms`, `rs`, `pd`, and `vi`) have a common syntax. Rather than restate this syntax for each command, we describe it below. (Note: This is the general syntax. Defaults and arguments definitions may differ between commands. See each command for details.) The syntax is:

command [*options*] [*mem_addr*] [*count*] [*format*]

where:

command The name of the command to be used

options One or more of the following options (multiple options may be grouped into one string with a preceding dash):

- `-p` Interpret the memory address as a physical address, not a logical address. This means the read or write physical routine will be called directly to read or write data.
- `-u` Specify that the memory address is in user space, not kernel space.
- `-l`*n* Specify that *n* elements can be printed across a line.
- `-v` Verify; see the the `memwrite` command description below.
- `n` Turn off converting labels to their symbolic form during printing memory locations.

mem_addr An expression specifying the starting address in memory to be examined; the default is the current *view_pc*.

count How many elements are to be operated on; the default is determined by the command itself.

format The format of the elements to be examined; the default is determined by the command itself.

Formats

The formats supported for the memory commands are as follows:

`decimal`	The memory location is a 16-bit decimal value. Aliases for decimal are `dec` and `d`.
`octal`	The memory location is a 16-bit octal value. Aliases for octal are `oct` and `o`.
`character`	The memory location is an 8-bit character value. Aliases for character are `char` and `c`.
`i`	The memory location is the start of an instruction.
`b`	The memory location is an 8-bit value in the default radix (octal, decimal, or hexadecimal). Set the default radix with the `mode` command.
`ld`	The memory location is a 32-bit decimal value.
`lo`	The memory location is a 32-bit octal value.
`lh`	The memory location is a 32-bit hexadecimal value.
`hex`	The memory location is a 32-bit hex value. Aliases for hex are `h` and `x`.
`s`	The memory location is a 16-bit value in the default radix (octal, decimal, or hexadecimal). Set the default radix with the `mode` command.
`l`	The memory location is a 32-bit value in the default radix (octal, decimal, or hexadecimal). Set the default radix with the `mode` command.
`ssym`	The memory location is a 16-bit symbolic value.
`sym`	The memory location is a 32-bit symbolic value.
`str`	The memory location is the start of a string, terminated by a null.
`pte`	The memory location is a page table entry.
`def`	If no format is specified, the default format is used. Each memory command has its own real default format.

The memory reference commands are listed below.

memread: Display Memory

The `memread` (or `mr`) command displays memory starting at the memory address, in the given format, for the specified number of elements. The number of elements displayed per line depends on the format selected. The current memory address is displayed at the beginning of the line. The default format is `long` and the default count is 1.

memwrite: View and modify memory

The `memwrite` (or `mw`) command allows the user both to view and modify memory locations one at a time. The modification starts at the memory address. It will continue until either count elements have been displayed or a `q` has been entered. Memory elements must be modified in the format specified. The default format is `long` and the default count is 1.

Memory write displays the element at the memory address in the format specified followed by a right angle bracket (>). You may then enter a response to that value. The

valid responses are listed below.

`q`	Exit `memwrite`.
`^`	Leave this location untouched, but display the previous element for modification.
`NL,CR`	Leave this location untouched, but display the next element for modification.
expression	Resolve the expression, expecting the format specified and write the results into the memory location. If the verify flag is set, redisplay this element, otherwise display the next element for modification.

In short, you are allowed to scan through memory modifying it selectively. Please note that modifying instructions is allowed, but this may affect the content of the next instruction.

memsearch: Search Memory

The `memsearch` (or `ms`) command searches through memory for a given value in a given format. The search starts at memory address and continues for a maximum of count elements. The default count is 1 and the default format is `long`. You will be prompted for the search value. You must enter the search value in the format specified. If a value matching the search value is found, a view (see `view` command) is performed at the location where the match occurred.

regsearch: Regular Expression Search

The `regsearch` (or `rs`) command is essentially the result of piping the output of an `mr` command through a regular expression filter. You will be prompted for the search value. All command line arguments are identical to that of `mr`.

patdump: Pattern Dump

The `patdump` (or `pd`) command has a similar interface to the `mr` command. The `pd` command takes a regular expression rather than an expression for the memory address. It then searches the symbol table for all matches to the regular expression. When found, the `mr` command is called using the arguments to the `pd` command with the symbol found replacing the regular expression. This command allow the user to look at a group of locations that can be described with a regular expression. This is useful for dumping a set of meters, counts, etc., that have a similar name.

view: View Memory in Code Format

The `view` (or `vi`) command is similar to the `memread` command, but it displays elements in a different format. The view command is used to display the element at the memory address surrounded by six elements on either side of the memory address. This is useful when looking at code in instruction mode and wanting to see the neighboring instructions. The default format is instruction mode. The count argument is not used and may be ignored. The default memory address is the view_pc. If a memory_address is given to the view command, that memory address becomes the new view_pc.

down: View Down

The `down` (or `do`) command increments the view_pc such that sequential executions of this command will produce a continuous listing of elements. The syntax for this command is as follows:

```
down
```

up: View Up

The `up` (view up) command decrements the view_pc such that sequential executions of this command will produce a continuous listing of elements. The syntax for this command is as follows:

```
up
```

ts: Translate an Expression Value to a Symbol

The `ts` command evaluates the expression given and converts it to a symbolic value using the current set of symbol tables. If a relevant symbol cannot be found, the value is converted based on the current radix. The resulting string is then printed. The syntax for this command is as follows:

`ts` [*expression*]

nm: Print Symbol Table Entry

The `nm` command searches the symbol table for a match to the regular expression given. If a match is found, the symbol is printed along with its value and symbol type. The syntax for this command is as follows:

`nm` [*regular_expression*]

General Commands

eval: Evaluate Expressions

The `eval` (or `ev`) command evaluates the expression given and prints the result in octal, decimal, and symbol formats. The syntax for this command is as follows:

`eval` [*expression*]

print: Print Values

The `print` command evaluates the contents of the memory address or user-defined variable, and prints the value of the contents in the current output radix. The syntax is as follows:

`print` [*expression*]

mode: Set Default Radix

The `mode` command sets the default radix (octal, decimal, hexadecimal) for the `short` and `long` memory command options, and turns editread (er) on or off. The syntax is as follows:

`mode` [`oct` | `dec` | `hex`] [`er` {`on`|`off`}]

global

The syntax for the `global` command follows:

`global` [*global_name*] [*expression*]

or

`global -d` [*global_name* ...]

The `global` command enables you to create a 32-bit variable that can be used in expressions. This allows you to save values for later use. A global variable will override the evaluation of a symbol of the same name. If a *global_name* is not given, the current list of *global_names* with their values is printed. You can initialize the global variable to a value by specifying an expression as the second argument. Global variables can be deleted with the `-d` option.

set

The syntax for the `set` command follows:

`set` [*global_name*] [*expression*]

The `set` command allows you to set a global variable to the evaluation of an expression.

help

The `help` command prints help information about a command. If the `help` command is invoked with no arguments, the list of supported commands is printed separated by subsystem. If a command name is given, the help string associated with that command is printed. The syntax for this command is as follows:

`help` [*command_name*]

Debugger Commands

b: Set or List Breakpoints

The `brk` command (same as `b`, `br` or `bp`) is used to set and list breakpoints in the kernel debugger. Since `crash` cannot be run on live kernels, its `brk` command is a no-op.

delete: Delete Breakpoints

The `delete` command is used to delete breakpoints in the kernel debugger. Since `crash` cannot be run on live kernels, its `delete` command is a no-op.

proceed: Proceed from a Breakpoint

The `b` command is used to continue execution from a breakpoint kernel debugger. Since `crash` cannot be run on live kernels, its `proceed` command is a no-op.

trace: Trace Back through a Process's Stack

The `trace` command will display a traceback of a process's kernel and/or user stack. If a process index is not given, the currently bound process will be traced. If a process index is given, that process will be traced. The syntax for this command is as follows:

`trace` [*options*] [*process_index*]

register: Display the Value of a General Register

The `register` (or `reg`) command with no arguments displays the values of all 32 general registers. With one argument (either an integer in the range 0-31, or the letter "r" followed by such an integer), `reg` displays the value of the indicated register only.

control: Display the Value of a Control Register

The `control` (or `ctl`) command with no arguments displays the values of all 19 control registers. With one argument (either the letters "cr" followed by such an integer in the range 2 through 20 or the mnemonic name of a control register), `ctl` displays the value of the indicated register only.

cmmu: Display the Contents of the CMMUs

The `cmmu` command causes the contents of the 88200 Cache and Memory Management Units to be displayed, if they are available.

ltop: Convert a logical address to a physical address

The `ltop` command converts the given logical address to a physical address and prints the result. The address space is assumed to be the one currently bound.

status: Display DG/UX System Information

The `stat` command displays useful information about the DG/UX system. The syntax for this command is as follows:

```
stat
```

vp: Describe a VP

The `vp` command displays the state of the specified virtual processor state block. If no vp_id is given, states for all of the VP state blocks will be displayed. The syntax for this command is as follows:

`vp` [*vp_id*]

focus: Look at the Address Space of a Given Process

The `focus` (or `fo`) command allows the user to look at the address space of the given process. The process is selected by process index. This allows the user to look at the per-process and user state of that process. The syntax for this command is as follows:

`focus` [*options*] [*process_index*]

halt: Halt the System

The `halt` command takes no arguments and is used in the kernel debugger to halt system execution and return to the SCM. Since `crash` can not be run on live kernels, its `halt` command is equivalent to `quit` and will cause the `crash` program to terminate.

Support Commands

quit: Exit from crash

The `quit` command is used to exit the program. This command will call the `exit`(2) system call with a 0 status code. The syntax for this command is as follows:

```
quit
```

fh: Print File Header Information

The `fh` command prints information obtained from the file header of the kernel being debugged. It takes no arguments.

x: Execute an External Macro File

The `x` command takes a UNIX filename as an argument. It reads the contents of that file one line at a time, executing each line as if it were a command line typed into `crash`.

Expressions

This section describes valid expressions. An expression cannot contain any white space. The expression is an arithmetic expression that results in one value. The elements of the expression are symbols, integers, the value of memory locations, binary operators, and unary operators. Parentheses can be used.

The binary operators perform an operation on two values. For example *v1 op v2*. The valid binary operators are:

- `+` Add *v1* and *v2*.
- `-` Subtract *v2* from *v1*.
- `*` Multiply *v1* by *v2*.
- `/` Divide *v1* by *v2*.

`&` Logical `and`.

`|` Logical `or`.

`>` 1 if *v1* > *v2* and 0 if *v1* <= *v2*.

`>=` 1 if *v1* >= *v2* and 0 if *v1* < *v2*.

`<` 1 if *v1* < *v2* and 0 if *v1* >= *v2*.

`<=` 1 if *v1* <= *v2* and 0 if *v1* > *v2*.

`=` 1 if *v1* = *v2* and 0 if *v1* != *v2*.

The unary operators perform an operation on a single value. For example *op v1*. The valid unary operators are:

`#` Read the 32-bit value at the address *v1*.

`@` Read the 16-bit value at the address *v1*.

`!` Logical NOT.

`:` Translate user space address to kernel space address.

The values used in the expression come from:

Symbol table values.
Integers.
Expression evaluations.
Addresses.

Symbols are resolved in the order listed as follows:

Debugger symbol values — These can be system constants or even machine state. The user sets up the definition of these symbols of the expression evaluation routines. For the 88k debugger, the 32 general-purpose registers r0-r31 and the control registers are available for use in expressions.

User-defined values — The user sets up the values of a global variable.

Kernel symbol values — Kernel symbols correspond to the names of C language routines, functions, and global variables in the kernel code.

Integers — These are integer constants in either decimal, octal, or hex representation.

Expression evaluations

FILES

`/dev/mem` Default system image file
`/dgux` Default namelist file

SEE ALSO

`crash`(8).

NAME

cron – clock agent

SYNOPSIS

/etc/cron

DESCRIPTION

Cron executes commands at specified dates and times. You can schedule commands on a regular basis according to instructions found in crontab files; crontab files are submitted via the crontab command. You may also schedule commands which are to be executed only once via the at command.

NOTE: You must have permission to create crontab files.

Commands that are to be executed only once may be submitted via the at command. Because cron never exits, it should be executed only once. This is best done by running cron from the initialization process through the rc script mechanism. (see rc.init(1M)).

To keep a log of all actions taken by cron, CRONLOG=YES (by default) must be specified in the /etc/default/cron file. If CRONLOG=NO is specified, no logging is done. Keeping the log is a user configurable option since cron can potentially create huge log files.

You can change the way cron schedules jobs by changing entries in the queuedefs file. The file has two lines, one for the at queue (a.) and one for the batch queue (b.):

a.*X*j*Y*n*Z*w
b.*X*j*Y*n*Z*w

where:

X is the maximum number of jobs allowed to execute simultaneously. This cannot be set higher than 25.

Y is the nice factor—the job priority number will be raised by this amount. The higher the number, the less attention the job gets from the CPU. The maximum nice factor is 20.

Z is how long to wait, in seconds, before trying to reschedule a queue request when the job queue is full.

The default limits are set to work as follows:

```
a.4j1n
b.2j2n90w
```

If you increase the job limits, be on guard for a potential impact on system performance, especially on smaller machines or machines having lots of active users.

FILES

/etc/cron.d	main cron directory
/etc/cron.d/queuedefs	scheduling information
/var/cron/log	accounting information (log file)
/var/spool/cron	spool area
/etc/default/cron	defaults file

DIAGNOSTICS

A history of all actions taken by cron is recorded in /var/spool/cron/log.

SEE ALSO

at(1), crontab(1), init(1M), sh(1).

NOTES

Cron(1M) examines crontab files (located in /var/spool/cron/crontabs) and at command files (located in /var/spool/cron/atjobs) only during process initialization and when a file changes. This reduces the overhead of checking for new or changed files.

BUGS

When it runs out of jobs to do, cron(1M) tries to redo jobs it has already done. This behavior is potentially dangerous, so you should always keep it busy, preferably with something like uuclean or a dummy job. This bug originated in AT&T System V.

NAME

dbm - general dbm(3X) database management tool

SYNOPSIS

dbm [-AILRSU] [-d *dbm_file*] [-m *mode*] [-o *output_file*] *command* [*args*...]

DESCRIPTION

Dbm is used to manage dbm(3X) type databases. Its function is controlled by the dbm *command* given on the command line, possibly with additional arguments. Its typical usage is to load a dbm database from an input file or to dump it to a readable format. It may also be used to probe for selected keys or add/delete specific key/value pairs. In addition, dbm provides parsing routines for an extended textual format, suitable for building tables of various kinds. The format is further described in the section about the parse command below.

Options

-A Append mode, don't automatically clear the database on the load and make operations.

-I Insert mode; will supply the DBM_INSERT flag to all *dbm_store* operations. This means that if two entries with equal keys are given, only the first will actually be entered to the database and no warning will be given. The default, if neither -I nor -R is supplied and two entries with equal keys are given, is to enter the first into the database and print a warning about the second.

-L Lowercase mode. Change all keys to lowercase before reading from or writing to the database.

-R Replace mode; will supply the DBM_REPLACE flag to all *dbm_store* operations. This means that if two entries with equal keys are given, only the last will actually be entered to the database and no warning messages are given.

-S Add a @@@ sentinel after the last entry has been written to the database.

-U Uppercase mode. Turn all keys to uppercase before reading from or writing to the database.

-d *dbm_file*
Perform all operations on the named database file. If no -d option is given, the last argument after the *command* will be used as the *dbm_file*.

-m *mode*
Use the given *mode* - an octal integer, such as 0644, when creating new databases. Not applicable if reloading an existing database.

-o *output_file*
Send all output from the dump and parse operations to the named output file instead of stdout.

Commands

clear Creates an empty dbm database, either by clearing an old one or by creating a new.

delete key [...]
Removes entries with the specified keys from the database.

dump Dumps the dbm database to stdout (or to *output_file*, if the -o option is used). The output will consist of one entry per line with a tab between each key and value.

fetch *key* [...]
Fetch will search for the specified keys in the database and print in dump

format on the standard output both key and value if found. Non-existing *keys* will be signaled by a [NOT_FOUND] message.

`load` [*file* . . .]
Load the database with entries from the specified *files*. If no *files* are given or if a file is specified as '-', the database will be loaded from standard input. Each line of the file should have a key and value separated by a tab. (Incidentally, this is the same format as the `dump` command will produce.) The database is first cleared unless the append (`-A`) switch has been given.

`make` [*file* . . .]
`Make` combines the operations of `parse` and `load` (q.v.), by storing each record after it has been parsed.

`parse` [*file* . . .]
This command will parse the contents of the specified *files* (or stdin if no *files* are given or when a file is '-'), according to the following syntax:

value key key . . .

Whitespace delimit tokens and sharp signs (#) anywhere on a line begins comments unless any of them are quoted by a backslash (\) or put inside double quotes ("...") or angle brackets (<...>). Lines beginning with whitespace are considered to be continuations of the previous line. Note that multiple keys for a given value are legal.

`store` *key value* [*key value* . . .]
Store one or more key/value pairs explicitly mentioned on the command line.

EXAMPLES

`dbm -d foo clear`	*create the database foo*
`cat infile | dbm load foo`	*load it from the infile*
`dbm parse xfile | dbm -AI load foo`	*add keys from the xfile...*
(or, shorter)	*...not already present...*
`dbm -I make xfile foo`	*...in the database*
`dbm fetch keya keyb foo`	*fetch values for the keys*
`dbm -R store keyc valuec foo`	*overwrite previous keyc value*
`dbm delete keyd valued`	*delete keyd and valued*

SEE ALSO

`sendmail`(1M), `dbm`(3X), `ndbm`(3C).

BUGS

Should probably remove the sentinel when opening the database for write access (provided that the `-S` flag has been given).

NAME

devattr - lists device attributes

SYNOPSIS

devattr [-v] *device* [*attribute* [. . .]]

DESCRIPTION

Devattr displays the values for a device's attributes. The display can be presented in two formats. Used without the -v option, only the attribute values are shown. Used with the -v option, the attributes are shown in an *attribute=value* format. When no attributes are given on the command line, all attributes for the specified device are displayed in alphabetical order by attribute name. If attributes are given on the command line, only those are shown and they are displayed in command line order.

The options and arguments for this command are:

-v	Specifies verbose format. Attribute values are displayed in an *attribute=value* format.
device	Defines the device whose attributes should be displayed. Can be the pathname of the device or the device alias.
attribute	Defines which attribute, or attributes, should be shown. Default is to show all attributes for a device. See the putdev(1M) manual page for a complete listing and description of available attributes.

DIAGNOSTICS

The command will exit with one of the following values:

0 = successful completion of the task.

1 = command syntax incorrect, invalid option used, or internal error occurred.

2 = device table could not be opened for reading.

3 = requested device could not be found in the device table.

4 = requested attribute not defined for specified device.

FILES

/etc/device.tab

SEE ALSO

getdev(1M), putdev(1M).

NAME

devfree – release devices from exclusive use

SYNOPSIS

devfree *key* [*device* [. . .]]

DESCRIPTION

Devfree releases devices from exclusive use. Exclusive use is requested with the command devreserv.

When devfree is invoked with only the *key* argument, it releases all devices that have been reserved for that *key*. When called with *key* and *device* arguments, devfree releases the specified devices that have been reserved with that *key*.

The arguments for this command are:

key Designates the unique key on which the device was reserved.

device Defines device that this command will release from exclusive use. Can be the pathname of the device or the device alias.

DIAGNOSTICS

The command will exit with one of the following values:

0 = successful completion of the task.

1 = command syntax incorrect, invalid option used, or internal error occurred.

2 = device table or device reservation table could not be opened for reading.

3 = reservation release could not be completely fulfilled because one or more of the devices was not reserved or was not reserved on the specified key.

FILES

/etc/device.tab
/etc/devlkfile

NOTES

The commands devreserv and devfree are used to manage the availability of devices on a system. These commands do not place any constraints on the access to the device. They serve only as a centralized bookkeeping point for those who wish to use them. Processes that do not use devreserv may concurrently use a device with a process that has reserved that device.

SEE ALSO

devattr(1M), devreserv(1M), getdev(1M), putdev(1M).

NAME

devnm – device name

SYNOPSIS

/etc/devnm [*name* ...]

DESCRIPTION

Devnm identifies the special file associated with the mounted file system where the argument *name* resides. Argument names must be full pathnames.

This command could be used, for example, to construct a mount table entry for the root device.

EXAMPLE

In the DG/UX System the command:

```
/etc/devnm /usr
```

produces

```
/dev/dsk/usr /usr
```

if /dev/dsk/usr is mounted as /usr.

FILES

/dev/dsk/*
/etc/mnttab

SEE ALSO

setmnt(1M), mnttab(4).

NAME

devreserv – reserve devices for exclusive use

SYNOPSIS

devreserv [*key* [*devicelist* [. . .]]]

DESCRIPTION

devreserv reserves devices for exclusive use. When the device is no longer required, use devfree to release it.

devreserv reserves at most one device per *devicelist*. Each list is searched in linear order until the first available device is found. If a device cannot be reserved from each list, the entire reservation fails.

When devreserv is invoked without arguments, it lists the devices that are currently reserved and shows to which key it was reserved. When devreserv is invoked with only the *key* argument, it lists the devices that are currently reserved to that key.

The arguments for this command are:

key Designates a unique key on which the device will be reserved. The key must be a positive integer.

devicelist Defines a list of devices that devreserv will search to find an available device. (The list must be formatted as a single argument to the shell.)

EXAMPLE

To reserve a floppy disk and a cartridge tape:

```
$ key=$$
$ echo "The current Process ID is equal to: $key"
The Current Process ID is equal to: 10658
$ devreserv $key diskette1 ctape1
```

To list all devices currently reserved:

```
$ devreserv
disk1          2423
diskette1      10658
ctape1         10658
```

To list all devices currently reserved to a particular key:

```
$ devreserv $key
diskette1
ctape1
```

DIAGNOSTICS

The command will exit with one of the following values:

0 = successful completion of the task.

1 = command syntax incorrect, invalid option used, or internal error occurred.

2 = device table or device reservation table could not be opened for reading.

3 = device reservation request could not be fulfilled.

FILES

/etc/device.tab
/etc/devlkfile

NOTES

The commands devreserv and devfree are used to manage the availability of

devices on a system. Their use is on a participatory basis and they do not place any constraints on the actual access to the device. They serve as a centralized bookkeeping point for those who wish to use them. To summarize, devices which have been reserved cannot be used by processes which utilize the device reservation functions until the reservation has been canceled. However, processes that do not use device reservation may use a device that has been reserved since such a process would not have checked for its reservation status.

SEE ALSO

devattr(1M), devfree(1M), getdev(1M), putdev(1M).

NAME

df – report number of free disk blocks and inodes

SYNOPSIS

df [-F *fs_type*] [-bcefgiklntV] [-o *options*] [*file-sys* | *special* | *directory*] ...

where:

fs_type	dg/ux or nfs
options	One or more options (see description of -o below)
file-sys	The name of the mount-point directory of a file system (e.g., /usr)
special	A special filename referring to a device containing a file system (e.g., /dev/dsk/usr)
directory	A directory name

DESCRIPTION

Df displays the number of free blocks and free file entries (inodes) available for online file systems. If you specify a directory, df gives information for the file system containing that directory. If you omit the arguments, df shows the free space on all of the mounted file systems.

Options

-b Report only the number of free kilobytes available on the file system.

-c If directory is a CPD or is contained within a CPD, compute free blocks and free inodes based on the CPD limits rather than on the file system limits.

-e Report only the number of free file entries on the file system.

-f Count only the blocks in the free list (free file entries are not reported). With this option, df can report on raw devices.

-g Report information returned by the statvfs(2) function call for the file system(s).

-i Report on inode use using a Berkeley-style format which includes the count of used and free inodes and the percent of inodes in use.

-k Report on free space using a Berkeley-style format which includes total space, space in use, space available for non-super-users, and the percent of the non-super-user capacity currently in use. The sizes are in kilobytes.

-l Report only on local file systems.

-n Report only the type of the file system.

-o Provide *fs_type*-specific options. This option is ignored in the DG/UX System, but is allowed for compatibility with other systems.

-t Report also the total number of blocks and file entries initially allocated to the file system. The -t option can be used together with the -k or -i options to report the totals for all requested file systems.

-F Report only on file systems of type *fs_type*.

-V Echo the complete command line, but do not execute the command. The command line is generated by using the options and arguments provided by the user and adding to them information derived from /etc/mnttab. This option should be used to verify and validate the command line.

The sum of the space in use and the available space reported by the -k option may be less than the total space because some space (typically around 10%) is reserved for allocation by the super-user. The capacity figure is based on the space available for a

non-super-user. Because it is possible for the super-user to allocate space after the non-super-user's space is full, it is possible for more than 100 percent of capacity to be reported, and it is possible for free space to be negative.

The numbers of free blocks and free file nodes take into account any applicable control point directory limits.

Inode information and control point directory limits are not available for remote file systems. Unavailable information is indicated by a question mark or a zero value.

EXAMPLES

```
df /usr
df /dev/dsk/usr
df /dev/rdsk/usr
```

The first example specifies a mounted directory name.
The second and third examples specify device names in the DG/UX System.

FILES

/dev/dsk/*

SEE ALSO

cpd(1), fs(4), mnttab(4).

NAME

dg_fsdb - file system debugger

SYNOPSIS

/etc/dg_fsdb [*special* | *mount-point*]

where:

special The block special device containing the file system

mount-point
A directory for which there is an entry in the /etc/fstab file, indicating the location where the file system device is mounted

DESCRIPTION

Dg_fsdb(1M) views information on a local file system. This information includes inodes, directory entries, and any other file system information.

To use dg_fsdb(1M) effectively, you must be familiar with the contents of inodes and directory entries and how these structures are used.

Dg_fsdb(1M) can display file system information in the following formats:

inode
directory entry
long
short
byte
character

Dg_fsdb(1M) supports decimal, hexadecimal, and octal numbers. A zero followed by a lower-case x indicates a hexadecimal number and a zero prefix indicates an octal number. Decimal is the default. Additionally, numbers can have units by suffixing the value with the following: B (blocks), I (node-number), K (kilobytes), M (megabytes), or C (bytes and the default).

Command Language

Unlike fsdb(1M), dg_fsdb(1M) provides a shell-like command language to walk through the directory hierarchy to display file system information.

The following commands are supported by dg_fsdb(1M):

help [*command-name*]
With no arguments, the names of all available commands are printed. If a *command-name* is given, a synopsis, syntax, and description of the command are printed.

open [*special* | *mount-point*]
Opens a file system for manipulation. If a command line argument is used to specify a file system, then a file system will already be opened. File systems may be opened at any time. *Special* is the block special device containing the file system. *Mount-point* is a directory for which there is an entry in the /etc/fstab file indicating the location the file system device should be mounted.

cd [*node-number* | *path*]
When dg_fsdb(1M) is started with a file system argument or a file system has been opened, the current directory is set to the root of the file system. cd provides the ability to change this directory. If no arguments are given, the path defaults to the root directory of the file system.

`pwd`
Prints the current working directory.

`ls` [*-ldc*] [*node-number* | *path*]
Prints a listing of the given directory specified by *node-number* or *path*. If no arguments are given, information about the current directory is printed.

The `-l` option specifies a long listing.

The `-d` option lists the directory's inode rather than its contents.

The `-c` option gives a complete listing of the inode beyond what is supplied by the `-l` option.

`env`
Prints statistics about the currently open file system.

`da` *disk-address*
Displays information about the given disk-address. A disk address contains no spaces and is of the form:

[*path* | *value*] [*:sequence-number*] [*+data-address*]

The first optional part provides the ability to specify an inode or absolute displacement in the file system. An inode can be specified with a path or an inode number value. Inode number values are suffixed with an I. If the value does not represent an inode, the value is assumed to be an absolute displacement into the file system. If the first part is not provided, the inode number of the current directory is used.

The second optional part allows the reference of a sequence number. This should only be used when the first part of the address refers to a directory. A sequence number of 0 refers to the directory entry in the given inode's parent directory (this provides an easy mechanism for looking at directory entries for a given path). If no value is given, the address will not resolve to a directory entry.

The third part provides the ability to access the data associated with a given inode. The *data-address* is a displacement within the file expressed by the given inode or default.

`dr` *disk-address* [*count*] [*format*]
Disk read reads and displays the contents of the given disk address. The *count* indicates the number of items to be printed starting from the given disk-address. The item type is specified with the *format* argument. The following formats can be used:

long,l
short,s
byte,b
character,char,c
decimal,dec,d
longdec,ld,D
octal,oct,o
longoct,lo,O

hexadecimal,hex,h,x
longhex,lhex,lh,H,X
string,str

`source` *path*
Reads and executes commands from the given *path*. The commands are executed on the current environment.

`alias` [*alias-name cmd-name arguments...*]
Replaces the old command with the new command and its arguments. If no arguments are given, all of the current aliases are printed.

`exit`
Exits `dg_fsdb`(1M)

If a .fsdbrc file exists in the user's home directory, this file is sourced before the program prompts the user for commands.

EXAMPLES

The following is an example session of `dg_fsdb`(1M):

```
fsdb> open /dev/dsk/test
fsdb> env
File System: /dev/dsk/test1
Size:   3200          Blocks
        1638400       Bytes
        1.562500      Megabytes
        0.001526      Gigabytes
Inodes: 4032
Current Directory: /
fsdb> ls
fooa(3)    foob(4)
fsdb> ls -l
   3 f---rw-rw-r--  1  510   50  1376256 Apr  9 10:37 fooa
   4 d---rwxrwxr-x  1  510   50  1376256 Apr  9 10:37 foob
fsdb> cd 4
fsdb> cd ../foob
fsdb> pwd
/foob(4)
fsdb> dr /foob+3B 0x1 c
/foob+0600> 0
fsdb> dr 4I+2K 010 H
3I+0800> 30313233 34353637 38394142 43444546 30313233
3I+0814> 34353637 38394142 43444546
fsdb> exit
```

SEE ALSO

`fsck`(1M), `fsdb`(1M), `fs`(4), `inode`(4).

NAME

dg_sysctl - display or modify boot and dump parameters

SYNOPSIS

dg_sysctl [-t] [-R] [-r *reboot-state*] [-b *boot-path*]
[-d *auto-dump-state*] [-f *dump-device*] [-l *dump-level*]
[-p *poweroff-state*]

DESCRIPTION

Use dg_sysctl to display or modify these boot and dump parameters: auto-boot behavior, boot path, auto-dump behavior, dump device, dump level, and auto-poweroff behavior. Entered without options, dg_sysctl displays the current settings. While any user can display values, only the super-user can change any.

The valid options are:

-t By default, changes you make with dg_sysctl remain in effect permanently, or until you change them. Use this option to make the changes temporary: after the next reboot, changed values revert to whatever they were before the change.

If you use this option, it must be the first option specified on the command line, and it must be followed by one or more of the options explained below.

-R Reset defaults: set auto-boot (-r) to halt, auto-dump (-d) to ask, dump level (-l) to kernel, and discard any previous boot path (-b) or dump device (-f) changes made with dg_sysctl(1M).

This option may be preceded by the -t option. No options other than -t are permitted on the command line.

-r halt | auto
Set the reboot behavior. The default is halt: after a panic, the systems halts and waits to be rebooted manually. If set to auto, the system tries to reboot after a panic, using the current boot path.

-b boot-path
Specify the SCM boot pathname, enclosed in quotes. The default boot path is that used for the most recent boot. (If you specify an empty name or spaces, the default SCM boot path is used.)

-d ask | skip | auto
Specify system behavior after a panic. The default is ask: after a panic, a prompt appears asking whether you want to dump memory contents to tape. If you specify skip, the system gives you no opportunity to dump memory to tape. If you specify auto, the system trys to dump memory contents without asking.

Note that, if auto is set and a panic occurs:

(1) Any tape in the dump device that is not write-protected will be overwritten.
(2) If the dump device contains no tape, or a write-protected tape, or a tape that is too small to hold the memory dump, no opportunity is provided to restart the dump with a new tape.

-f dump-device
Specify the dump device to use after a panic. The default is the value of the

DUMP variable configured in the current kernel. Enclose the device name in quotes, and enter the name in DG/UX common device specification format. (Devices and naming conventions are explained in *Managing the DG/UX System*, Appendix A.)

`-l kernel | all`
Specify which main memory frames to dump during a memory dump: either `kernel` frames—the default—or `all` memory frames.

Note: unless requested by Data General to change this setting, leave it set to the default. Kernel-frame dumps are smaller and faster, and usually contain all the information needed to understand the cause of a panic.

`-p auto | skip`
Specify poweroff behavior after a normal shutdown. If you specify `auto`, the system attempts to power itself off after a normal shutdown. If you specify `skip`, the system does not attempt to power itself off.

This setting applies only to normal shutdowns. It has no affect on system behavior after a panic or other abnormal shutdown.

Not all systems support automatic poweroff. On systems which support automatic poweroff, the default value is `auto`; on other systems, the default is `skip`.

EXAMPLE

```
dg_sysctl -r auto -b "/dgux -3" -d auto -f "st(insc(),4)" -l all
```
Enable auto-reboot after a panic; reboot the kernel `/dgux` to init level 3; enable the auto-dump after a panic; dump memory contents to SCSI tape device 4; and dump all memory frames.

FILES

`/etc/default/dg_sysctl`
Stores the parameters that have been set by `dg_sysctl`. This file does not exist unless permanent changes have been made using `dg_sysctl`.

DIAGNOSTICS

The `dg_sysctl` exit codes have the following meanings:

0 The operation was successful.

1 The operation was unsuccessful.

2 The operation failed due to access restrictions.

3 There was an error in the command line.

SEE ALSO

`reboot`(1M), `dg_sysctl`(2), *Using the AViiON® System Control Monitor(SCM)* (014-001802). *Managing the DG/UX System* (093-701088).

NAME

dg_telnetd - TELNET protocol server

SYNOPSIS

/usr/bin/dg_telnetd [-l *file*] [-d] [-c] [-s] [-m] [-u] [-p *psterm*]

DESCRIPTION

The dg_telnetd server supports the DARPA standard TELNET virtual terminal protocol. The TELNET server is invoked by the inetd server when an incoming connection is detected on the port specified in /etc/services. See inetd(1M) and services(4) for details.

The dg_telnetd server operates by allocating a pseudoterminal device (see pty(7)) for a client, then creating a login process that has the slave side of the pseudoterminal as stdin, stdout, and stderr. The dg_telnetd server manipulates the master side of the pseudoterminal, implementing the TELNET protocol and passing characters between the client and login process.

By default, dg_telnetd will search for pseudoterminals named /dev/ptyp*N*, where *N* starts at 0 and goes to 255. This search can be changed with the -p option, which specifies the first name in a sequence of names to search. You can use the -p switch up to three times to start the search. There is usually no need to use the -p option.

Use the -d option to enable debugging. Actions of dg_telnetd invoked with this option are undefined. However, in general, dg_telnetd writes additional information regarding its actions to the logfile.

The -l option and *file* name will write dg_telnetd log information to that file.

When you use the -c option, the daemon initially negotiates remote echoing and suppresses the Go-Ahead option. If the client program negotiates local echoing, the option is turned off. In DG/UX TCP/IP Revision 5.4 this is the default mode of operation.

When you use the -s option, the daemon dynamically switches between line at a time and character at a time modes of operation based on the needs of the application program. This was the default mode of operation for dg_telnetd for Revisions prior to DG/UX TCP/IP 5.4

The -m option works like -c, but it cannot be turned off by any option negotiation sequence.

Use the -u option to prevent the daemon from sending any data marked as urgent. This was previously accomplished through the -c option.

SEE ALSO

inetd(1M), telnet(1C), pty(7).

NAME

diskman – menu interface for managing physical and logical disks

SYNOPSIS

diskman [*option*]

DESCRIPTION

Only the superuser can execute the diskman command.

The DG/UX system comes with two versions of the diskman program:

stand-alone You invoke this version directly from tape when you are installing the DG/UX system, or when your system is down, you can boot the diskman disk image at the SCM> prompt. The disk image, located on your /usr logical disk, is /stand/diskman.

stand-among You can invoke this version through the sysadm diskmgmt command or directly from the command line.

The diskman program contains a complete set of menu-driven procedures for creating and managing your physical and logical disks. When invoked without an option, diskman presents a menu of disk management selections, from which the user may select.

For a complete explanation of the diskman program, see the chapter "Disk Management" in *Managing the DG/UX System*. The information on this reference page is meant to serve as an outline of diskman's functionality.

Options

You can invoke diskman from the command line with the following options. Physical disks must be specified in DG/UX common format, such as cied(,0). Logical disks are specified by name, such as binky1. Note that physical disk specifications contain characters that are special to the shell, so such arguments should be enclosed in quotes.

display_registered_disks	To display all registered physical disks
register_disk *disk_specification*	To register a physical disk
deregister_disk *disk_specification*	To deregister a physical disk
display_ld_info *logical_disk_name*	To display logical disk information

diskman Menus

Typing diskman without an option displays the Main Menu. See "Disk Management" in *Managing the DG/UX System* for details.

SEE ALSO

sysadm(1M).

NAME

diskusg - generate disk accounting data by user id

SYNOPSIS

/usr/lib/acct/diskusg [-svipu] [*file* ...]

where:

file The name of a file system or of a file containing output from a previous invocation of diskusg; the default is the standard input.

DESCRIPTION

Diskusg generates disk accounting information. If you omit the -s option, you must specify a file system name. Diskusg then extracts the disk accounting information from the inodes of the specified file systems. With -s you must specify a diskusg output filename.

Diskusg produces one line per user, in the following format:

uid login #blocks

where:

uid The numerical user id of the user
login The login name of the user; and
#blocks The total number of disk blocks allocated to this user.

Options

-s Assemble data from existing diskusg output files, combining all lines for a single user into a single line.

-v Be verbose, printing on standard error a list of all files charged to no one. Diskusg prints 'BAD UID' followed by a file system name, cylinder group number, inode number, and user identification number.

-i *fnmlist* Ignore the data on those file systems whose file system name is in *fnmlist* . *Fnmlist* is a list of file system names separated by commas or enclosed within quotes. Diskusg compares each name in this list with the file system name stored in the volume ID (see *labelit(1M)).*

-p *file* Use *file* instead of /etc/passwd as the name of the password file to generate login names.

-u *file* Write records to *file* of files that are charged to no one. Records consist of the special file name, the i-node number, the user ID, and the number of blocks.

The output of diskusg is normally the input to acctdisk (see acct(1M)), which generates total accounting records that can be merged with other accounting records. Diskusg is normally run in dodisk (see acctsh(1M)).

FILES

/etc/passwd Used for user id to login name conversions

SEE ALSO

acct(1M), acctsh(1M), acct(4).

NOTE

Diskusg cannot be used to generate accounting information for remotely mounted filesystems.

NAME

dkctl - control special disk operations

SYNOPSIS

dkctl [-t] *special* [*option*]

where:

special Pathname of a DG/UX block or character special physical disk device
option Option to enable or disable on the disk

DESCRIPTION

The dkctl command is used to enable or disable special disk options. In particular the enabling or disabling of verified writes (write check functionality) is controlled by this command.

Options are:

-t Make the changes temporary. The changes will not be preserved across system reboot.

wchk Enable write checking for the physical disk specified by *special*. This means that all writes to the physical disk will be explicitly verified to have been correctly written on the disk. Write verify can be used to enhance data integrity, but will result in degraded disk write performance. This is the default state for physical disks using optical media.

-wchk Disable write checking for the physical disk specified by *special*. This is the default state for all other physical disk devices.

EXAMPLES

```
dkctl /dev/rpdsk/0 wchk
dkctl /dev/pdsk/1 -wchk
```

FILES

/etc/default/dkctl
Stores the default values for the dkctl parameters.

DIAGNOSTICS

Exit status is 0 if successful, 1 if an error occurs.

SEE ALSO

sd(7), cied(7), cimd(7).

NOTE

Use of the dkctl command requires super-user permissions.

NAME

dump - incremental file system dump

SYNOPSIS

/usr/sbin/dump [*key* [*argument* ...] *special*]

DESCRIPTION

The dump command copies to magnetic tape all files changed after a certain date in a particular file system. (You may prefer to use the dump2(1M) command rather than dump because dump2 is faster; otherwise, there is no disadvantage to using dump.) *Special* is the pathname of a special file referring to a device containing a file system. *Key* specifies the date and other options about the dump. The key consists of characters from the set 0123456789bcdfgJnsuWwz:

0-9 Indicate the dump level. All files modified since the last date stored in the file /etc/dumpdates for the same file system at lesser levels will be dumped. If no date is determined by the level, the beginning of Jan. 1, 1970, GMT, is assumed; thus the option 0 dumps the entire file system.

b Specify blocking factor, the number of 1024-byte blocks per tape record. Default is 10; maximum is 32. Ideally, this number will match the optimal blocking factor for the tape device.

g Specify a memory buffer size expressed as 1K blocks. Default is set to the value of the -b option (or 10, if -b is not used). The maximum is 2048. The buffer must be at least the size and a multiple of the -b value and may also be limited by memory available. Note that increasing this buffer will allow you to stream devices.

c The tape used is a cartridge tape. Dump(1M) considers this factor when it determines how much it can write on one tape. See also the s option.

d Take the density of the tape, expressed in bits per inch(bpi), from the next *argument*. This is used in calculating how much can be written to each tape. The default is 1600 bpi.

f Place the dump on the next *argument* file instead of the tape. If you have DG TCP/IP (DG/UX), you can use this option to dump to a remote device. For example,

```
dump 0f sys:/dev/rmt/0 /root
```

lets you dump the root filesystem to the tape device "0" on system "sys." To do this, you must be logged in as root on your own system, and your system must have an entry in the remote host's /.rhosts file.

J Convert the old, obsolete format to the new format. All other options are ignored, and dump terminates immediately. Invoke this option only when the old /etc/ddate files are updated to the new /etc/dumpdates format.

n Notify an operator (as in wall(1M)) whenever a response is required at the operator's console. /etc/group must contain an entry for "operator."

s Specify the size of the dump tape in feet. The number of feet is taken from the next *argument*. When the specified size is reached, dump waits for the tape to be changed. The default tape size is 2300 feet. Type of tape is also a factor in dump's calculation of tape length; see the c option.

u Write the date of the beginning of the dump on file /etc/dumpdates, if the dump completes successfully. This file records a separate date for each file

system and each dump level. You can read the format of /etc/dumpdates, which consists of one free format record per line: file system name, increment level, and ctime(3)-format dump date. You can edit /etc/dumpdates to change any of the fields. Note that /etc/dumpdates is formatted differently from previous versions of dump in /etc/ddate, although it contains identical information. This option may cause errors if your /etc/dumpdates file contains entries generated by the dump2(1M) command. See the NOTES section.

W Tell the operator what file systems need to be dumped. This information is gleaned from the files /etc/dumpdates and /etc/fstab. Dump prints out the most recent dump date and level for each file system in /etc/dumpdates, and highlights those file systems that should be dumped. All other options are ignored, and dump exits immediately. This option may cause errors if your /etc/dumpdates file contains entries generated by the dump2(1M) command. See the NOTES section.

w Do as W does, but print only those file systems that need to be dumped. This option may cause errors if your /etc/dumpdates file contains entries generated by the dump2(1M) command. See the NOTES section.

z Print the inode numbers of dumped files on the standard output.

If no arguments are given, *key* is assumed to be 9u and a default file system is dumped to the default tape.

Dump and restore support symbolic links and control point directories.

Dump requires operator intervention on end of tape, end of dump, tape write error, tape open error, or disk read error (if there are more than 32 errors). In addition to alerting all operators (with the n key), dump interacts with the operator on dump's control terminal when dump can no longer proceed, or if something is grossly wrong. All questions dump poses must be answered by typing yes or no.

Since making a full dump involves a lot of time and effort, dump checkpoints itself at the start of each tape volume. If writing that volume fails for some reason, dump will, with operator permission, restart itself from the checkpoint after the old tape has been rewound and removed, and a new tape has been mounted.

At periodic intervals, dump tells the operator what is going on, usually including low estimates of the number of blocks to write, the number of tapes it will take, the time to completion, and the time to the tape change. The output is verbose, so that others know that the terminal controlling dump is busy and will be for some time.

To perform dumps, start with a full level 0 dump:

```
dump 0un
```

Next, dumps of active file systems are taken on a daily basis, using a modified Tower of Hanoi algorithm, with this sequence of dump levels:

3 2 5 4 7 6 9 8 9 ...

For the daily dumps, a set of 10 tapes per dumped file system is used on a cyclical basis. Each week, a level 1 dump is taken, and the daily Hanoi sequence repeats with 3. For weekly dumps, a set of 5 tapes per dumped file system is used, also on a cyclical basis. Each month a level 0 dump, which is saved indefinitely, is taken on a set of fresh tapes.

FILES

/etc/dumpdates	New format dump date record
/etc/fstab	Dump table: file systems and frequency

/etc/group To find group *operator*

DIAGNOSTICS

Many, and verbose.

SEE ALSO

`restore`(1M), `fstab`(4).

NOTES

The `dump` command will return an error if you invoke it with the `u`, `W`, or `w` options after the `/etc/dumpdates` file has been written to with a `dump2 -u` command (see `dump2`(1M)). Before using `dump` with the `u`, `W`, or `w` options in such cases, you should first remove all entries from the `/etc/dumpdates` file.

BUGS

Sizes are based on 1600 BPI blocked tape. Fewer than 32 read errors on the file system are ignored. Since each tape requires a new process, parent processes for tapes already written continue until the entire tape is written.

`Dump` should know about the dump sequence, keep track of the tapes used, tell the operator which tape to mount and when, and provide more help to the operator running `restore`.

NAME

dump2 - incremental file system backup

SYNOPSIS

/usr/sbin/dump2 [*options*] *file-system*

DESCRIPTION

Dump2 creates a data file of all files changed after a certain date in a particular file system. *file-system* is the pathname of a special file referring to a device containing a file system. Note that *file-system* must refer to a local file system (not a file system mounted from another host).

File system dumps created with dump2 can be read by restore(1M).

Options

-dump-level

Indicate the dump level (0 through 9). All files modified since the last date stored in the file /etc/dumpdates for the same file system at lesser levels will be dumped. If no date is determined by the level, the beginning of Jan. 1, 1970, GMT, is assumed; thus the *dump-level* 0 dumps the entire file system. The default is 9.

-B *number-of-buffers*

Specify *number-of-buffers* as the number of shared memory buffers to use. A larger number may increase the speed of dumps. The default is 3.

-b *buffer-size*

Specify *buffer-size* to be the number of 1024-byte blocks written per record. For tape devices which require blocking factors, this argument should match the optimal blocking factor for the particular device in use. The default is 10; the maximum is 64.

-D *output-disk-file-name*

Write the output to *output-disk-file-name* without any tape headers or trailers. This output can be used as input for another dump by specifying the -T option.

-E *exclude-list-file*

Read the file *exclude-list-file* for a list of inode numbers to exclude from the dump. Each inode number must appear on a separate line. Any characters after the inode number are ignored. The *exclude-list-file* may be "-" to indicate that standard input is to be read.

-f *dump-device*

Place the dump on *dump-device*. The default is /dev/rmt/0. If you have DG TCP/IP (DG/UX), you can use this option to dump to a remote device. For example,

```
dump2 -0 -f sys:/dev/rmt/0 /dev/rdsk/root
```

lets you dump the root filesystem to the tape device "0" on system "sys." To do this, you must be logged in as root on your own system, and your system must have an entry in the remote host's /.rhosts file.

-I *include-list-file*

Read the file *include-list-file* for a list of inode numbers to include in the dump. Each inode number must appear on a separate line. Any characters after the inode number are ignored. The *include-list-file* may be "-" to indicate that standard input is to be read.

`-i` Ignore tape size estimates. This allows `dump2` to write to the physical end of tape, rather than stopping when the estimate indicates that end of tape is near.

`-L` *log-file-name*
Specify *log-file-name* as the name of the log file in which to record messages from `dump2`. If this option is used, all messages normally written to standard error are also appended to *log-file-name*. If this option is not used, messages are written only to standard error.

`-M` *medium-name*
Specify *medium-name* as the type of medium being dumped to. *medium-name* must be an entry in the tape table file (see the `-t` option).

`-n` Notify an operator (as in `wall`(1M)) whenever a response is required at the operator's console. `/etc/group` must contain an entry for "operator".

`-O` *operator-input-fifo*
Read operator input (i.e. answers to queries) from a fifo-special file. Normally, `dump2` writes operator messages to stderr and reads operator responses from /dev/tty. This option causes operator input to be read from *operator-input-fifo* instead. This is very useful when running `dump2` from `cron`(1M) since `cron` jobs have no controlling tty. For example,

```
dump2 -0 -f /dev/rmt/0 -O /tmp/fifo /dev/rdsk/root 2>/dev/console
```

would send all output of dump2 to the console, but would solicit operator responses from /tmp/fifo. Running this command from a cron job would allow you to send `dump2` output to the console without having to take control of the console for input. Operator queries from `dump2` (such as requests for the next tape) could be answered by echoing the strings "yes" or "no" to /tmp/fifo.

`-s` Report performance statistics after completion of dump. The performance report includes the total elapsed time subdivided into: time spent in initialization, time spent actually dumping data, time spent waiting on tapes to rewind (not including rewind time for the final tape), and time spent waiting on operator intervention. It also includes the total amount of data dumped and the average data transfer rate for the dump.

`-T` *input-file-name*
Read filesystem information from *input-file-name*. This file must be in the form produced by running `dump2` with the `-D` option.

`-t` *tape-table-file-name*
Read medium information from *tape-table-file-name*. The default is `/etc/dumptab`.

`-u` Write the date of the beginning of the dump on the file `/etc/dumpdates`, if the dump completes successfully. This file records a separate date for each file system and each dump level. The `/etc/dumpdates` file consists of one free format record per line: file system name, increment level, and `ctime`-format dump date.

`-z` Print the inode numbers of dumped files on the standard output.

`dump2` and `restore` support symbolic links and control point directories.

dump2 requires operator intervention on end of tape, end of dump, tape write error, tape open error, or disk read error (if there are more than 32 errors). In addition to alerting all operators (with the -n option), dump2 interacts with the operator on the dump2 command's control terminal when dump2 can no longer proceed, or if something is grossly wrong. All questions dump2 poses must be answered by typing yes or no.

Because making a full dump involves a lot of time and effort, dump2 allows the dump to continue if a bad tape block is encountered. If at any point dump2 fails to write to the tape, dump2 will prompt the operator for a new tape, and continue the dump.

At periodic intervals, dump2 tells the operator what is going on, usually including low estimates of the number of blocks to write, the number of tapes it will take, the time to completion, and the time to the tape change. The output is verbose, so that others know that the terminal controlling dump2 is busy and will be for some time.

Inode Lists

dump2 generates a list of inodes which are to be backed up. This list includes all inodes which have changed since the last lower-level dump, and which appear in some *include-list-file,* and which do not appear in any *exclude-list-file*. By default, the *include-list-file* contains all inodes for the file system, and the *exclude-list-file* contains no inodes.

The inode numbers needed for either the *include-list-file* or the *exclude-list-file* can be generated from the -i option to the ls(1) command, or from the st_ino field of the stat(5) structure.

Dump Cycles

To perform dumps, start with a full level 0 dump:

```
dump2 -0un /dev/rdsk/root
```

Next, dumps of active file systems are taken on a daily basis, using a modified Tower of Hanoi algorithm, with this sequence of dump levels:

```
3 2 5 4 7 6 9 8 9 9 ...
```

For the daily dumps, a set of 10 tapes per dumped file system is used on a cyclical basis. Each week, a level 1 dump is taken, and the daily Hanoi sequence repeats with 3. For weekly dumps, a set of 5 tapes per dumped file system is used, also on a cyclical basis. Each month a level 0 dump, which is saved indefinitely, is taken on a set of fresh tapes.

EXAMPLES

In order to perform a complete backup of the root file system, invoke dump2 as follows:

```
dump2 -0 -f /dev/rmt/0 /dev/dsk/root
```

In order to archive all files the "/home" file system belonging to user "smith", use the following lines:

```
find /home -user smith -exec ls -id {} \; > /tmp/file.list
dump2 -I /tmp/file.list -0 -f /dev/rmt/0 /home
rm /tmp/file.list
```

FILES

`/etc/dumpdates`	Previous dump dates for each file system
`/etc/fstab`	Dump frequency for each file system
`/etc/group`	Group entry for "operator"
`/etc/dumptab`	Table specifying media characteristics

SEE ALSO

`restore`(1M), `ctime`(3C), `dumptab`(4), `fstab`(4), and `group`(4).

NOTES

`Dump2` uses the `-B` and `-b` options to request approximately *number-of-buffers* * *buffer-size* * 1024 bytes of shared memory. If `dump2` cannot get this amount of shared memory, either or both of these arguments should be decreased. Alternatively, the system can be reconfigured to make more shared memory available.

NAME

dump2label - read and write labels for dump tapes

SYNOPSIS

/usr/sbin/dump2label [-f *tape-device*] [-l] [-n] [-p *file-number*]

DESCRIPTION

dump2label creates and extracts information from dump tape headers. This tool is designed to be used by sysadm(1M) and admbackup(1M) to track multiple file system dumps on a single tape.

A dump tape header consists of a list of text strings identifying the dumps that are to be placed on a particular tape.

Options

-f *tape-device*
: Read or write from *tape-device*. The default is /dev/rmt/0n.

-l
: List the dump tape header for this tape to the standard output.

-n
: Create a new tape header with labels read from standard input. Each line of input should be a text string which could later be used to identify the files on the tape.

-p *file-number*
: Position to file *file-number* on the tape. The first file after the tape label is considered file 1.

One of -n, -l, and -p must be specified on the command line.

Multiple-tape Dumps

dump2label prints the message "Next tape" and exits with status *n* (where *n* is the number of files remaining to be skipped) if the requested file is not on this tape. The user should mount the next tape, and invoke dump2label again to continue the search.

For example, suppose we want to position to the third of a series of dumps. The dump2label command line is

```
dump2label -f/dev/rmt/0n -p3
```

If only the first and second dumps are found on the first tape, dump2label will print "Next tape" and exit with status 1 (because one more file needs to be skipped). The user must then mount the next tape and invoke dump2label as

```
dump2label -f/dev/rmt/0n -p1
```

to position to the desired file.

EXIT STATUS

dump2label will exit with status 0 if it was successful. If an error occurs, the exit status will be -1. A positive exit status means that dump2label was unable to position to the requested file. In this case, the exit status is the number of files yet to be skipped.

NOTES

dump2label assumes that if the end-of-tape condition is true, the current file must continue on the next tape. This assumption is false if a file ends exactly at end-of-tape. Though this case is unlikely, if it does occur, dump2label returns a value that is one greater than it should be.

SEE ALSO

admbackup(1M), dump(1M), dump2(1M), sysadm(1M).

NAME

`dumpfs` – dump file system information

SYNOPSIS

`/usr/sbin/dumpfs` *filesys* | *special*

where:

filesys The pathname of a directory in `/etc/fstab`. If this argument is present, the device associated with that pathname will be processed.

special The pathname of a special file for a device on which a file system resides

DESCRIPTION

`Dumpfs` lists all super-block and disk allocation region (DAR) information for a file system. A DAR is an allocation grouping used to distribute files over a disk. It consists of three parts. The first two parts are of a fixed size. The first fixed portion is the bitmap of allocated data blocks in the DAR. The second fixed portion is the inode table for the allocation group. The last portion of the DAR is the actual space used for the files of the file system.

There are three sections of output by `dumpfs`:

1) superblock-related information
2) summary contents of the DAR entry table (DARE)
3) information for each DAR

Superblock Information

The following is printed:

- File system ID (fsid)
- minor device number
- DAR size
- number of inodes per DAR
- whether the file system is mountable
- file system name (fname)
- fpack (filesystem pack name)
- data element size
- index element size
- directory data element size
- directory index element size
- 1st and 2nd anniversaries (facilities that control the allocation of disk resources for each file in the filesystem)
- number of DARs
- number of usable blocks in the file system
- actual number of blocks in file system
- number of allocated inodes
- number of free inodes
- number of data blocks

DARE Table Information

The following is printed:

- number of allocated inodes
- number of allocated data blocks
- number of directories in the DAR

DAR Information

The following is printed:

- number of allocated inodes
- number of directories in the DAR
- list of free data blocks
- number of allocated data blocks
- histogram of the free data block sizes in the DAR

When summing up the sizes of the free data blocks, the bitmap is scanned starting with the bits representing the data area. The file system block allocation algorithms do not count any sequence of "data element size" free bits in the bitmap as a free data element. Contiguous data blocks will only be allocated as a data element if the sequence of bits in the bitmap are aligned on a "data element size" boundary. For example, if the data element size of the file system is 16 blocks, a data element will only be allocated if a sequence of 16 free bits in the bitmap are aligned on a 16-bit (word) boundary in the bitmap. This holds true for data element sizes up to 32 blocks, but any data element size greater than 32 blocks will search starting on any 32-bit (long word) boundary.

SEE ALSO

`fsck`(1M), `mkfs`(1M), `tunefs`(1M), `fs`(4), `fstab`(4).

NAME

exportfs - export and unexport directories to NFS clients

SYNOPSIS

/usr/etc/exportfs [-avui] [-o *options*] [directory]

DESCRIPTION

exportfs makes a local directory (or file) available for mounting over the network by NFS clients. It is normally invoked at boot time by the rc scripts, and uses information contained in the /etc/exports file to export a *directory* (which must be specified as a full pathname). The super-user can run exportfs at any time to alter the list or characteristics of exported directories. Directories that are currently exported are listed in the file /etc/xtab.

With no options or arguments, exportfs prints out the list of directories currently exported.

OPTIONS

-a All. Export all directories listed in /etc/exports, or if -u is specified, unexport all of the currently exported directories.

-v Verbose. Print each directory as it is exported or unexported.

-u Unexport the indicated directories.

-i Ignore the options in /etc/exports. Normally, exportfs will consult /etc/exports for the options associated with the exported directory.

-o *options*

Specify a comma-separated list of optional characteristics for the directory being exported. *options* can be selected from among:

-secure

Require clients to use a more secure protocol when accessing the directory.

NOTE: Secure RPC using DES Authentication is an additional feature that must be purchased separately from the DG/UX™ ONC™/NFS® package. You must have this feature to use the -secure option.

ro Export the directory read-only. If not specified, the directory is exported read-write.

rw=*hostname*[:*hostname*]...

Export the directory read-mostly. Read-mostly means exported read-only to most machines, but read-write to those specified. If not specified, the directory is exported read-write to all.

anon=*uid*

If a request comes from an unknown user, use *uid* as the effective user ID. Note: root users (uid 0) are always considered "unknown" by the NFS server, unless they are included in the "root" option below. The default value for this option is -2. Setting the value of "anon" to -1 disables anonymous access.

root=*hostname*[:*hostname*]...

Give root access only to the root users from a specified *hostname*. The default is for no hosts to be granted root access.

`access=`*client*[`:`*client*]...
Give mount access to each *client* listed. A *client* can either be a hostname, or a netgroup (see `netgroup`(4)). Each *client* in the list is first checked for in the `/etc/netgroup` database, and then the `/etc/hosts` database. The default value allows any machine to mount the given directory.

FILES

`/etc/exports`	static export information
`/etc/xtab`	current state of exported directories
`/etc/netgroup`	

SEE ALSO

`exports`(4), `netgroup`(4)

CAUTIONS

You cannot export a directory that is either a parent- or a sub-directory of one that is currently exported and *within the same filesystem*. It would be illegal, for example, to export both `/usr` and `/usr/local` if both directories resided in the same disk partition.

NAME

filesave, tapesave - daily/weekly file system backup

SYNOPSIS

/etc/filesave

/etc/tapesave

DESCRIPTION

These shell scripts are provided as models. They provide a simple, interactive operator environment for file backup. Filesave is for daily disk-to-disk backup and tapesave is for weekly disk-to-tape backup.

FILES

/var/adm/log/filesave.log

SEE ALSO

shutdown(1M), volcopy(1M).

NAME

fingerd, in.fingerd - remote user information server

SYNOPSIS

in.fingerd

DESCRIPTION

fingerd implements the server side of the Name/Finger protocol, specified in RFC 742. The Name/Finger protocol provides a remote interface to programs which display information on system status and individual users. The protocol imposes little structure on the format of the exchange between client and server. The client provides a single command line to the finger server which returns a printable reply.

fingerd waits for connections on TCP port 79. Once connected it reads a single command line terminated by a <RETURN-LINE-FEED> which is passed to finger(1). fingerd closes its connections as soon as the output is finished.

If the line is null (only a RETURN-LINEFEED is sent) then finger returns a default report that lists all users logged into the system at that moment.

If a user name is specified (for instance, eric<RETURN-LINE-FEED>) then the response lists more extended information for only that particular user, whether logged in or not. Allowable names in the command line include both login names and user names. If a name is ambiguous, all possible derivations are returned.

FILES

/var/utmp	who is logged in
/etc/passwd	for users' names
/etc/inetd.conf	enable or disable
/var/adm/lastlog	last login times
$HOME/.plan	plans
$HOME/.project	projects

SEE ALSO

finger(1)

Harrenstien, Ken, *NAME/FINGER*, RFC 742, Network Information Center, SRI International, Menlo Park, Calif., December 1977.

NOTES

fingerd is disabled by default. To enable fingerd, the administrator must uncomment the appropriate line from the /etc/inetd.conf configuration file.

Connecting directly to the server from a TIP or an equally narrow-minded TELNET-protocol user program can result in meaningless attempts at option negotiation being sent to the server, which will foul up the command line interpretation. fingerd should be taught to filter out IAC's and perhaps even respond negatively (IAC *will not)* to all option commands received.

NAME

frec – recover files from a backup tape

SYNOPSIS

/etc/frec [-p *path*] [-f *reqfile*] *raw-tape i-number*:*name* ...

where:

path	A default prefixing pathname different from the current working directory
reqfile	The name of a file that contains recovery requests of the format *i-number*:*newname*, one per line
raw-tape	The pathname of a tape drive on which a backup tape written by volcopy(1M) is mounted
i-number	The inode number of the file being recovered
name	The pathname of the file to which the data for each recovery request is written

DESCRIPTION

Frec recovers files from a specified backup tape. If a directory is missing in *name*, it is created.

Options are:

-p Prefix the specified pathname to any names that are not fully qualified, i.e., that do not begin with / or ./.

-f Read recovery requests from the specified file.

EXAMPLES

If a file with i-number 1216 has been backed up, you can recover the file into a file named junk in your current working directory by typing the following:

```
frec /dev/rmt/0m 1216:junk
```

To recover files with i-numbers 14156, 1232, and 3141 into files /usr/src/cmd/a, /usr/src/cmd/b, and /usr/joe/a.c:

```
frec -p /usr/src/cmd /dev/rmt/0m 14156:a 1232:b 3141:/usr/joe/a.c
```

SEE ALSO

cpio(1), volcopy(1M).

NOTES

While creating the intermediate directories contained in a pathname, frec can recover inode fields only for those directories contained on the tape and requested for recovery.

Frec expects the volume label to match on all tapes.

NAME

`fsck` - check file systems for consistency and repair them

SYNOPSIS

`/sbin/fsck` [`-l`] [`-y`] [`-n`] [`-p`] [`-q`] [`-x`] [`-t` *scratch_file*] [*special* ...]

DESCRIPTION

`Fsck` checks file systems and corrects inconsistencies. *special* is the pathname of a special file referring to a device containing a file system; the default is the standard set of special files listed in `/etc/checklist`. If no `/etc/checklist` exists, `/etc/fstab` is used.

If you wish to check several file systems at once, omit *special* and specify the `-p` option. The `/etc/fstab` file contains a pass number and a mounting status for each file system. When using this file, `fsck` checks only those file systems that have a non-zero pass number and a "rw" or "ro" mounting status. The order of file system checking is based on pass number, starting at 1.

To save time, have `fsck` check file systems on disks run by different controllers on the same pass. To do this, make the pass number for each of these file systems the same in `/etc/fstab`.

Invoking the fsck Program

There are two ways that `fsck` is invoked:

Startup script	This is the most common way of invoking `fsck`. When you are in multi-user mode bringing up the system with the `init` command, you can automatically execute `fsck` from within your startup script.
Command line	From the command line, you type: `fsck` [*options*] [*filesystem_names*].

Options

All options except for `-t` are boolean flags, and may thus be combined: `fsck -pxl`, for example. The following options are interpreted by `fsck`:

`-l` Perform fast recovery from the fsck log, if possible. Fast recovery can be performed only if the fsck_log_size option, which turns on fast recovery logging, was used the last time the file system was mounted and this is the first time that fsck has been run on the file system since that mount. If fast recovery cannot be performed on the file system, normal recovery will be.

Normal recovery causes the entire file system to be scanned during the consistency check. Fast recovery makes use of a log created while the system is running and the file system is mounted that details file system changes. By performing the consistency check only on the parts of the file system known to have changed, fsck in fast recovery mode runs in time proportional to the size of the fsck log rather than to the size of the file system.

`-p` Detect all possible inconsistencies, but correct only those inconsistencies that may be expected to occur from an abnormal system halt. For each corrected inconsistency, one or more lines will be printed identifying the file system and the nature of the correction. Any other inconsistencies will cause the check of that file system to fail. The following 15 inconsistencies (and only those listed) will be corrected for the specified file systems:

1. An inode has an incorrect count of the blocks it uses. The count is corrected.

2. An inode is partially truncated. This can occur if the system is abnormally halted while a file is being truncated, leaving it claiming more data blocks than its size in bytes would require. The extra blocks are freed.
3. A directory has an incorrect child count. The count is corrected.
4. A directory entry exists for an inode which is unallocated. The directory entry is removed.
5. A directory entry's filename length is incorrect. The length is corrected.
6. An inode is unreferenced (has no directory entries anywhere in the file system). The inode is reconnected in the `/lost+found` directory.
7. No `/lost+found` directory exists, but an inode needs to be reconnected there. The directory is created.
8. The root directory needs to be expanded in order to make room for a directory entry for `lost+found`. The directory is expanded.
9. The `/lost+found` directory needs to be expanded in order to make room for a directory entry for an inode being reconnected there. The directory is expanded.
10. An inode's link count is incorrect. The count is corrected.
11. The root control point directory's resource accounting (blocks, inodes) is incorrect. The counts are corrected.
12. A disk allocation region (DAR) has an incorrect free-block bitmap. The bitmap is corrected.
13. A DAR has an incorrect free-inode list. The list is corrected.
14. A DAR has incorrect summary counts of used blocks, inodes or directories. The counts are corrected.
15. The summary counts in the superblock are incorrect. The counts are corrected.

`-q` Repair the inconsistencies listed under the `-p` option automatically, without asking for user approval. Unlike `-p`, however, more serious inconsistencies will not cause `fsck` to fail; the user must still answer the resulting queries.

`-y` Audit and interactively repair all file system inconsistencies assuming a "yes" response to all questions asked by `fsck`. This option should be used with great caution, since it could lead to irreversible changes to the filesystem.

`-n` Audit and interactively repair all file system inconsistencies, assuming a "no" response to all questions asked by `fsck`. This option also means that all file systems will be opened with read-only intent.

`-x` File systems are examined before being checked. If a file system is marked mountable in its superblock, then it is not checked.

`-t` Use the specified scratch file for temporary storage if `fsck` cannot obtain enough memory. The scratch filename must be the next argument after `-t`.

The following options are mutually exclusive, and use of more than one per invocation is not allowed: `-y`, `-n`, `-p`, `-q`.

Checking

Checking proceeds without any input from the operator if no errors are discovered. When a fatal inconsistency is discovered, no further checking is done on that file system; `fsck` either exits or proceeds to the next specified file system. When an

inconsistency is discovered with the `-p` option, and that error is one of those listed under `-p`, the inconsistency if fixed without operator approval. Any other discoveries of inconsistencies require the operator to make a decision. The `fsck` program prompts with its recommended action. If you answer `yes`, then `fsck` takes the recommended action. In no case will any damaging action be taken without approval. Note, however, that advance approval or disapproval may be given by invoking `fsck` with the `-y` and `-n` options, respectively.

The `fsck` program checks for the following inconsistencies. Note that the term "Bad format" refers to system blocks that do not have the required self-identification information.

- Unreadable or inconsistent superblocks.
- Bad format in superblocks.
- Invalid contents in superblock's reserved area.
- Bad value for superblock's file system size.
- Bad value for superblock's DAR size.
- Bad value for superblock's inode/DAR density.
- Bad value for superblock's default data element size.
- Bad value for superblock's default index element size.
- Bad value for superblock's default directory data element size.
- Bad value for superblock's default directory index element size.
- Bad value for superblock's default first allocation threshold.
- Bad value for superblock's default second allocation threshold.
- Bad format in inode table block.
- Invalid contents in inode's reserved area.
- Files of unknown type.
- Files with bad fragment size.
- Files which are partially truncated.
- Files claiming impossible blocks.
- Files claiming system-area blocks.
- Bad Index-block format.
- Files with incorrect block counts.
- Files claiming already-claimed blocks.
- Unallocated root inode.
- Bad file type for root.
- Incorrect resource limit information in root.
- Incorrect parent directory in root.
- Directories with "holes" (unallocated blocks before end-of-file).
- Bad format in directory blocks.
- Directories with invalid information in reserved areas.

- Directories with empty blocks at end.
- Directories with incorrect child counts.
- Extra directory entries named "." or "..".
- Directory entries with invalid characters in filenames: "/" or non-ASCII characters.
- Directory entries whose pathnames are too lengthy.
- Directory entries that are out of order.
- Directory entries with incorrect entry lengths.
- Directory entries with incorrect filename lengths.
- Extraneous hard links to directories (including cycles in file system name space).
- Extraneous hard links to symbolic link files.
- Directory entries to invalid inodes.
- Directory entries to unallocated inodes.
- Files with incorrect space parent.
- Unconnected files or directories.
- Bad or missing `lost+found` directories.
- Bad `lost+found` directory entries.
- Root or `lost+found` directories needing expansion.
- Files with incorrect link counts.
- Incorrect resource allocation counts in control point directories.
- Bad format in DAR blocks.
- Invalid contents in reserved area of DAR blocks.
- Incorrect free-block bitmaps in DARs.
- Incorrect or incomplete free-inode lists in DARs.
- Incorrect DAR summary counts: blocks used, inodes used, directories used.
- Incorrect superblock summary counts.

Orphaned files and directories (allocated but unreferenced) are, with the operator's concurrence, reconnected by placing them in the `lost+found` directory. The name assigned is the (decimal) inode number preceded by the `#` character.

Checking the character-special device (`/dev/rdsk/*`) is almost always faster then checking the block-special device (`/dev/dsk/*`).

FILES

`/etc/fstab` Default list of file systems to check

SEE ALSO

`crash`(1M), `mkfs`(1M), `mount`(1M), `checklist`(4), `fs`(4), `fstab`(4), `crash`(8). *Installing the DG/UX System*, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NOTES

`Fsck` works sequentially from the pass numbers listed in `/etc/fstab`, that is, it begins at 1 and then runs all other passes in order. There is no way to start an `fsck -p` at pass *n*, where *n* is an arbitrary number.

`Fsck` support for `/etc/checklist` has been included only for compatibility purposes. We recommend that you do not have a `checklist` file.

NAME

`fsdb` – file system debugger

SYNOPSIS

`/etc/fsdb` *special* [`-`]

where:

special The pathname of a special file referring to a device containing a file system

DESCRIPTION

`Fsdb` views and alters information on the disk. This information includes inodes, directory entries, and any other disk information.

The `-` or an `0` argument suppresses error-checking routines that verify inode, directory, and block addresses. To do these checks, `fsdb` reads the *fs_nodes_per_dar* (inodes per disk allocation region) and *fs_size* (file system size) fields of of the file system's super-block.

`Fsdb` is intended mainly for emergencies where `fsck`(1M) and `clri`(1M) are ineffective in dealing with problems on the disk. To use `fsdb` effectively, you must be familiar with the contents of inodes and directory entries and how these structures are used.

`Fsdb` can display disk information in the following formats:

inode
directory entry
double word
word
byte
character

`Fsdb` supports decimal and octal numbers. A zero prefix indicates an octal number. Decimal is the default.

Examining Inodes

To view an inode, type an integer followed by a lowercase `i`. `fsdb` then lists the specified inode's number, mode, link count, user ID, group ID, size, the fragment numbers stored in its direct and indirect block arrays, and the creation, last-modification, and last-access times. The default is the current inode. When you enter `fsdb`, the current inode is number 2, indicating the root directory. Otherwise the current inode is the last inode referenced.

To change the fields of the inode, use the mnemonics given for a particular field followed by an equal sign and the desired contents for that field. For example, `sz=1024` sets the file size for the current inode to 1024 bytes. The i-number of an inode cannot be changed.

The following mnemonics are used to change inodes:

`md`	Mode
`ln`	Link count
`uid`	User ID number
`gid`	Group ID number
`sz`	File size in bytes
`a #`	Data block numbers (0-12 for ordinary files, and 0-10 for non-ordinary files)
`at`	Time of last access

`mt`	Time of last modification
`ct`	Time of creation
`maj`	Major device number (special files only)
`min`	Minor device number (special files only)

For control point directories only:

Max space usage	Total amount of space allowed in this and subordinate directories.
Cur space usage	Current amount of space allowed in this directory and subordinates.
Max file node	Total number of inodes allowed in this and subordinate directories.
Cur file node	Current number of inodes allowed in this directory and subordinate directories.

An inode's first 10 data blocks are direct blocks; the next 3 are indirect. To print the contents of the data blocks, type an `f`. The `f` can be followed by a logical block number and a print option or by a print option alone. If you omit the number, block 0 is printed. The data in the specified block are printed according to the mode indicated by a print option. The print options are:

`d`	Print as directories.
`o`	Print as octal words.
`e`	Print as decimal words.
`c`	Print as characters.
`b`	Print as octal bytes.

The `f` command does not work on a special file (device) or on empty data blocks. (The pointer in the inode for an empty block in a file is 0.)

The `a` field of the inode is examined and changed in a manner different from other fields. To examine a data block entry, specify `a` followed by a logical block number and then `b`. The address of the array element within the inode for the logical block number is printed, followed by `A`. Then the octal and the decimal disk block number of the data block itself is printed.

Examining Directory Entries

To view a directory entry, type a `d` followed by an integer. A directory entry consists of an inode number, the name length, entry length, sequence number, and filename. Since directory entries are found in the data of a directory inode, `fsdb` displays the directory entry specified relative to the current inode. If the current inode is not a directory inode or there are fewer directory entries than the one you indicate, `fsdb` prints this message:

```
nonexistent block
Cannot scan directory
```

All fields of the directory entry except the inode number can be changed by giving the mnemonic for the field followed by an equal sign and then a value. For the name field, the character string following the equal sign should be enclosed in double quotation marks if the first character is not alphabetic; otherwise, double quotation marks are optional for this field. To alter the inode number for a directory entry, omit the mnemonic. The mnemonics are as follows:

`nl`	Name length in characters
`el`	Entry length – total byte count used by the entry
`nm`	Filename
`sq`	Sequence number

Use extreme care in changing names, name lengths, and entry lengths, because following entries can be corrupted or made unreachable.

Examining Other Data

Information can also be printed as double words, words, bytes, or characters. To view data in these forms, one can use `D`, `W`, or `B` for double words, words, or bytes, respectively. Used alone, these options print the current address in octal, followed by an indicator of the mode requested (`D.`, `W.`, or `B.`). The contents of the current location follow in octal and then in decimal. If the mode desired is preceded by a number, that number is taken as the address and becomes the current address.

To alter the contents of a double word, word, or byte, follow the address and mode indicator (`B`, `W`, or `D`) by an equal sign and a value. For example, `01006W=0177777` sets the contents of word `01006` to `0177777`.

Current Address

The contents of the current address can also be viewed by following the letter `p` with an `i` for inodes or one of the print options described above for use with the `f` command. If the `p` is followed by a number, that many elements (inodes, words, etc., depending on the print option used) are printed.

The current address is normalized to an appropriate boundary before printing begins. The current address advances with the printing and is left at the address of the last item printed.

The current address can be saved by using the `<` symbol. The symbol `>` restores the saved address. Only one address can be saved at any one time; a subsequent use of `<` replaces a previously saved address with the current address.

Commands and Symbols

Following is a summary of `fsdb` commands:

[*n*]`i`	Print inode *n*; current is the default.	
	d*n*	Print directory entry number *n*.
	p[*n*]*o* | `i`	Print the next *n* inodes according to option *o* or in inode format.
	`f`[*n*]*o*	Print logical data block *n* according to option *o*; block 0 is the default.
[*a*]`B`	Enter byte mode at address *a*; default is the current address.	
[*a*]`W`	Enter word mode at address *a*; default is the current address.	
[*a*]`D`	Enter double-word mode at address *a*; default is the current address.	
	`+`,`-`	Print next item, last item.
	`>`,`<`	Save current address, restore saved address.
	`O`	Set or unset error-checking toggle.
	`!`	Escape to the shell.
	`q`	Quit; terminate the program.
	`<NL>`	Print next item of the previously specified type.

The following symbols are used for assignment to inode and directory fields and also to locations directly:

=*n*	Assignment

=+*n*	Incremental assignment
=-*n*	Decremental assignment

You can use dots, tabs, and spaces as delimiters within commands to `fsdb`, but they are not necessary. A line with just a new-line character increases the current address by the size of the data type last printed. That is, the address is set to the next byte, word, double word, directory entry, or inode, letting you step through a region of the file system. A line with just a + or - takes you to the next or previous item, respectively.

Miscellaneous

`Fsdb` cannot print more than a block's worth of data at once because only one block of data is kept in memory at any one time. If a count of 0 is used with the `p` command, all entries to the end of the current block are printed. Because `fsdb` deals with only a block at a time, raw devices and block devices can be debugged.

All assignment operations result in immediate write-through of the current block. During any assignment operation, numbers are checked for a possible truncation error due to the size of the destination address. The message "`alignment`" is printed if such a truncation would have taken place if the assignment had been made.

To terminate output at any time, type the delete character.

EXAMPLES

`2i`	Prints the root inode of the current file system.
`386i`	Prints inode number 386 in inode format; this becomes the current inode.
`ln=4`	Changes the link count for the current inode to 4.
`ln=+1`	Increases the current inode's link count by 1.
`fc`	Prints, in ASCII, block zero of the file associated with the current inode.
`f2c`	Prints the third data block of the current inode as characters.
`2i.fd`	Prints the root inode's first data block in the form of directory entries. This example combines several operations on one command line; the same effect would occur if the '2i' were followed by 'fd' as a separate command.
`d2`	Prints the third directory entry in the current inode.
`d0=32`	Sets the inode number of the first directory entry of the current inode to 32.
`d5i.fc`	Changes the current inode to that associated with the sixth directory entry of the current inode; the first logical block of the file is then printed in ASCII.
`0B.p0o`	Prints the super-block of a DG/UX file system in octal.
`2i.a0b.d7=3`	Changes the i-number for the eighth directory slot in the root directory to 3.
`d7.nm="name"`	Changes the name field in the directory slot to "name."
`a2b.p0d`	Prints the third block of the current inode as directory entries.

`0140B` Produces the output

```
0140B.:      14 (12)
```

if location 0140 happens to contain 014.

SEE ALSO

`clri`(1M), `fsck`(1M).

NAME

`ftpd` – File Transfer Protocol server

SYNOPSIS

`/usr/bin/ftpd [ -d ] [ -l ] [ -t`*timeout* `]`

where:

timeout A time-out value in seconds

DESCRIPTION

The `ftpd` process is the DARPA Internet File Transfer Prototocol (FTP) server process. The server uses the Transmission Control Protocol (TCP) as its transport protocol. The FTP server is invoked by the `inetd` server when an incoming connection is detected on the port specified in `/etc/services`. See `inetd`(1M) and `services`(4) for details.

Options

`-d` Enable debugging, with output going to `/tmp/ftpd*`.

`-l` Log each FTP session to the system log. For details about the system log, see `syslog`(3C).

`-t` Set the inactivity time-out period to the value specified. By default, the FTP server does not time out an inactive session.

Requests

The FTP server currently supports the following requests; case is not distinguished.

Request	Description
ABOR	abort any transfer in progress
ACCT	specify account (ignored)
ALLO	allocate storage
APPE	append to a file
CDUP	change to the parent of the current working directory
CWD	change working directory
DELE	delete a file
HELP	give help information
LIST	give list of files in a directory (`ls -l`)
MKD	make a directory
MODE	specify data transfer *mode*
NLST	give list of names of files in directory (`ls`)
NOOP	do nothing
PAGE	specify a new page size
PASS	specify password
PASV	listen on a data port and wait for a connection
PORT	specify data connection port
PWD	print the current working directory
QUIT	terminate session
REIN	reinitialize server state
REST	restart the last aborted transfer
RETR	retrieve a file
RMD	remove a directory
RNFR	specify rename-from filename
RNTO	specify rename-to filename
SEOR	specify a new end-of-record delimiter
SITE	display any information specific to the remote system

STAT	display server's status
STOR	store a file
STOU	store a file under a unique name
STRU	specify data transfer *structure*
TYPE	specify data transfer *type*
USER	specify username
XCUP	change to parent of current working directory
XCWD	change working directory
XMKD	make a directory
XPWD	print the current working directory
XRMD	remove a directory

The `ftpd` process interprets filenames according to the "globbing" conventions used by `csh`(1). This allows you to use the metacharacters "*?[]{}~".

User Authentication Rules

The `ftpd` process authenticates users according to three rules:

1) The username must be in the password database, `/etc/passwd`, or, if you use the Network Information Service, it must be in the Network Information Service password database. If a password is required for a given username, it must be provided by the client process before any file operations can be performed.

2) If the username is `anonymous` or `ftp`, an anonymous `ftp` login must be specified in the password file (user `ftp`). In this case, a user is allowed to log in by specifying any password (by convention, this is given as the client hostname).

3) The username must not be in the `/etc/ftpd.deny` file. If the username is in this file, `ftp` access is denied to the user.

In the second case, `ftpd` takes special measures to restrict the client's access privileges. The server performs a `chroot`(1M) command to the home directory of the `ftp` user. So that system security is not breached, it is recommended that the `ftp` subtree be constructed with care. The following guidelines are recommended.

`~ftp`	Make the home directory owned by `ftp` and unwritable by anyone.
`~ftp/bin`	Make this directory owned by the superuser and unwritable by anyone. The program `ls`(1) must be present to support the list commands. This program should have mode 111.
`~ftp/etc`	Make this directory owned by the superuser and unwritable by anyone. The files `passwd`(4) and `group`(4) must be present for the `ls` command to work properly. These files should be mode 444.
`~ftp/pub`	Make this directory mode 777 and owned by `ftp`. Users should then put in this directory all files that are to be accessible through the anonymous account.

SEE ALSO

`ftp`(1C), `inetd`(1M), `ftpd.deny`(4).

BUGS

The server must run as the superuser to create sockets with privileged port numbers. It maintains an effective user ID of the logged-in user, reverting to the superuser only when binding addresses to sockets.

NAME

`fuser` - identify processes using a file or file structure

SYNOPSIS

`/etc/fuser [ -ku ]` *files* `[ - ] [[ -ku ]` *files*`]`

DESCRIPTION

`Fuser` lists the process ids of the processes using files. `Fuser` does not identify processes on remote machines using files through NFS. For block special devices, all processes using any file on that device are listed. The process ids are followed by `c` or `r` if the process is using the file as its current directory or its root directory and a `t` if the process is currently executing the file.

If you use the `-u` option, the login name, in parentheses, also follows the process id. In addition, if the `-k` option is specified, the SIGKILL signal is sent to each process. Only the superuser can terminate another user's process (see `kill`(2)). Options may be re-specified between groups of files. The new set of options replaces the old set; a single dash (-) cancels any options currently in force.

The process ids are printed as a single line on the standard output, separated by spaces and terminated with a single new line. All other output is written on standard error.

EXAMPLES

`/etc/fuser -ku /dev/dsk/test`
: If typed by the superuser, terminates all processes that are preventing the file system on the logical disk named 'test' from being unmounted and lists the process IDs and login name of each process as it is killed.

`/etc/fuser -u /etc/passwd`
: Lists process IDs and login names of processes that have the password file open.

`/etc/fuser -ku /dev/dsk/test -u /etc/passwd`
: Performs both of the first two examples in a single command line.

SEE ALSO

`mount`(1M), `ps`(1), `dg_file_info`(2), `dg_process_info`(2), `kill`(2), `signal`(2).

NAME

fwtmp, wtmpfix - manipulate connect accounting records

SYNOPSIS

```
/usr/lib/acct/fwtmp [ -ic ]
/usr/lib/acct/wtmpfix [files]
```

DESCRIPTION

Fwtmp

Fwtmp reads from the standard input and writes to the standard output, converting binary records of the type found in wtmp to formatted ASCII records. The ASCII version is useful to enable editing, via ed(1), bad records or general purpose maintenance of the file.

The argument -ic is used to denote that input is in ASCII form, and output is to be written in binary form.

Wtmpfix

Wtmpfix examines the standard input or named files in wtmp format, corrects the time/date stamps to make the entries consistent, and writes to the standard output. A - can be used in place of *files* to indicate the standard input. If time/date corrections are not performed, acctcon(1) will fault when it encounters certain date-change records.

Each time the date is set, a pair of date change records are written to /etc/wtmp. The first record is the old date denoted by the string old time placed in the line field and the flag OLD_TIME placed in the type field of the <utmp.h> structure. The second record specifies the new date and is denoted by the string new time placed in the line field and the flag NEW_TIME placed in the type field. *Wtmpfix* uses these records to synchronize all time stamps in the file.

In addition to correcting time/date stamps, wtmpfix will check the validity of the name field to ensure that it consists solely of alphanumeric characters or spaces. If it encounters a name that is considered invalid, it will change the login name to INVALID and write a diagnostic to the standard error. In this way, wtmpfix reduces the chance that acctcon(1) will fail when processing connect accounting records.

FILES

```
/etc/wtmp
/usr/include/utmp.h
```

SEE ALSO

acct(1M), acctcms(1M), acctcom(1), acctcon(1M), acctmerg(1M), acctprc(1M), acctsh(1M), runacct(1M), ed(1), acct(2), acct(4), utmp(4).

NAME

getdev - lists devices based on criteria

SYNOPSIS

getdev [-ae] [*criteria* ...] [*device* ...]

where:

criteria An expression defining criteria that a device must match to be included in the generated list

device A device pathname or device alias specifying which devices should be included in the generated list

DESCRIPTION

Getdev generates a list of devices that match certain criteria. The criteria includes a list of attributes (given in expressions) and a list of devices. If no criteria is given, all devices are included in the list.

Devices must satisfy at least one of the criteria in the list unless the -a option is used. Then, only those devices which match all of the criteria in a list will be included.

Devices which are defined on the command line and which match the criteria are included in the generated list. However, if the -e flag is used, the list becomes a set of devices to be *excluded* from the list.

Criteria Expression Types

There are four possible expression types which the criteria specified in the *criteria* argument may follow:

attribute=value Selects all devices whose attribute *attribute* is defined and is equal to *value*.

attribute!=*value* Selects all devices whose attribute *attribute* is defined and does not equal *value*.

attribute:* Selects all devices which have the attribute *attribute* defined.

attribute!:* Selects all devices which do not have the attribute *attribute* defined.

See the putdev(1M) manual page for a complete listing and description of available attributes.

Options

The options for this command are:

-a Specifies that a device must match all criteria to be included in the list generated by this command. The flag has no effect if no criteria are defined.

-e Specifies that the list of devices which follows on the command line should be *excluded* from the list generated by this command. (Without the -e the named devices are *included* in the generated list.) The flag has no effect if no devices are defined.

FILES

/etc/device.tab

DIAGNOSTICS

The command will exit with one of the following values:

0 = Successful completion of the task.

1 = Command syntax incorrect, invalid option used, or internal error occurred.

2 = Device table could not be opened for reading.

SEE ALSO

devattr(1M), getdgrp(1M), putdev(1M),

NAME

getdgrp – lists device groups which contain devices that match criteria

SYNOPSIS

getdgrp [-ael] [*criteria* [. . .]] [*dgroup* [. . .]]

where:

criteria Criteria that a device must match before a device group to which it belongs can be included in the generated list

dgroup Device groups which should be included in or excluded from the generated list.

DESCRIPTION

Getdgrp generates a list of device groups that contain devices matching the given criteria. The criteria is given in the form of expressions.

criteria can be one expression or a list of expressions which a device must meet for its group to be included in the list generated by getdgrp. If no criteria is given, all device groups are included in the list.

Devices must satisfy at least one of the criteria in the list. However, the -a flag can be used to define that a "logical and" operation should be performed. Then, only those groups containing devices which match all of the criteria in a list will be included.

dgroup defines a set of device groups to be included in the list. Device groups that are defined and which contain devices matching the criteria are included. However, if the -e flag is used, this list defines a set of device groups to be excluded. When the -e option is used and criteria is also defined, the generated list will include device groups containing devices which match the criteria and are not in the command line list.

Criteria Expression Types

There are four possible expressions types:

attribute=*value*	Selects all device groups with a member whose attribute *attribute* is defined and is equal to *value*.
attribute!=*value*	Selects all device groups with a member whose attribute *attribute* is defined and does not equal *value*.
attribute:*	Selects all device groups with a member which has the attribute *attribute* defined.
attribute!:*	Selects all device groups with a member which does not have the attribute *attribute* defined.

See the putdev(1M) manual page for a complete listing and description of available attributes.

Options

The options for this command are:

-a Specifies that a device must match all criteria before a device group to which it belongs can be included in the list generated by this command. The flag has no effect if no criteria are defined.

-e Specifies that the list of device groups on the command line should be *excluded* from the list generated by this command. (Without the -e the named device groups are the only ones which can be *included* in the generated list.) The flag has no effect if no device groups are defined.

`-l` Specifies that all device groups (subject to the `-e` option and the *dgroup* list) should be listed even if they contain no valid device members. This option has no affect if *criteria* is specified on the command line.

FILES

`/etc/device.tab`
`/etc/dgroup.tab`

DIAGNOSTICS

The command will exit with one of the following values:

`0` = successful completion of the task.

`1` = command syntax incorrect, invalid option used, or internal error occurred.

`2` = device table or device group table could not be opened for reading.

SEE ALSO

`devattr`(1M), `getdev`(1M), `listdgrp`(1M), `putdev`(1M), `putdgrp`(1M).

NAME

getmany – get MIB-classes from SNMP agent

SYNOPSIS

getmany *host community object ...*

where:

host is a hostname or Internet address
community is a community string
object is a class name

DESCRIPTION

Use the getmany command to retrieve classes of object instances or to walk the entire Management Information Base (MIB). The command sends an SNMP message containing the *community* string and a GetNextRequest-PDU requesting the specified *object* to the agent running on *host*. The command continues to query the agent with GetNextRequest-PDUs until either the end of the MIB is detected or the agent returns an object not in the current class.

Specify the *host* as either a hostname or an Internet address in dot-notation.

The *community* string is a text string used by the agent to authenticate the request. You must configure the agent to accept requests from the specified *community* for the operation to be successful.

The *object* is specified as an object class which may be either an object type, object instance, or group name. You can specify the *object* in either dot-notation representing an object identifier or as a text string representing an object descriptor.

EXAMPLES

The first example shows getmany using the object class system.

```
$ getmany myhost public system

Name: sysDescr.0
Value: DG/UX TCP/IP SNMP AGENT

Name: sysObjectID.0
Value: DataGeneral

Name: sysUpTime.0
Value: 1342175

Name: sysContact.0
Value: myname@myhost

Name: sysName.0
Value: myhost

Name: sysLocation.0
Value: Hall 106 Office 1

Name: sysServices.0
Value: 72
```

The second example shows how getmany could be used to retrieve the entire MIB.

```
$ getmany myhost public iso

Name: sysDescr.0
Value: DG/UX TCP/IP SNMP AGENT

Name: sysObjectID.0
Value: DataGeneral

.
.
.

Name: udpLocalPort.0.0.0.0.1212
Value: 1212

Name: udpLocalPort.0.0.0.0.2049
Value: 2049
End of MIB.
```

DIAGNOSTICS

Exit status is 0 upon success.

Exit status is −1 if there are errors parsing the command line.

Exit status is 1 if there is an error returned from the agent.

SEE ALSO

getone(1M), getnext(1M), setany(1M), snmpd(1M), trap_recv(1M), trap_send(1M), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

getnext - get MIB-objects from SNMP agent

SYNOPSIS

getnext *host community object ...*

where:

host is a hostname or Internet address
community is a community string
object is a object type or object instance

DESCRIPTION

Use the getnext command to retrieve objects from an SNMP agent. The command sends an SNMP message containing the *community* string and a GetNextRequest-PDU requesting the specified *object* to the agent running on the *host*.

Specify the *host* as either a hostname or an Internet address in dot-notation.

The *community* string is a text string used by the agent to authenticate the request. You must configure the agent to accept requests from the specified *community* for the operation to be successful.

Specify the *object* as either an object type or object instance. You may specify the *object* in either dot-notation representing an object identifier or as a text string representing an object descriptor.

EXAMPLES

```
$ getnext myhost public sysDescr sysDescr.0  1.3.6.1.2.1.2

Name: sysDescr.0
Value: DG/UX TCP/IP SNMP AGENT

Name: sysObjectID.0
Value: DataGeneral

Name: ifNumber.0
Value: 2
```

Note: The object returned by the agent is the one that follows lexicographically the one specified.
In this example the type sysDescr was specified and the first instance sysDescr.0 was returned.
The instance sysDescr.0 was specified so the next instance sysObjectID.0 was returned.
Lastly, 1.3.6.1.2.1.2 which is the object identifier for ifTable was requested and ifNumber.0, which is the first instance in the ifTable was returned.

DIAGNOSTICS

Exit status is 0 upon success.

Exit status is -1 if there are errors parsing the command line.

Exit status is 1 if the agent returns an error.

SEE ALSO

getone(1M), getmany(1M), setany(1M), snmpd(1M), trap_recv(1M), trap_send(1M), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

getone – get MIB-object from SNMP agent

SYNOPSIS

getone *host community object ...*

where:

host is a hostname or Internet address
community is a community string
object is a fully qualified object name

DESCRIPTION

Use the getone command to retrieve object instances from an SNMP agent. The getone command sends an SNMP message containing the *community* string and a GetRequest-PDU requesting the specified *object* from the agent running on *host*.

Specify the *host* as either a hostname or as an Internet address in dot-notation.

The *community* string is a text string used by the agent to authenticate the request. You must configure the agent to accept requests from the specified *community* for the operation to be successful.

The *object* specified must be fully qualified, which means that it should represent an object instance instead of an object type. For example, sysDescr represents an object type while sysDescr.0 represents an object instance. You may specify the *object* in either dot-notation representing the object identifier or as a text string representing the object descriptor.

EXAMPLE

```
$ getone myhost public sysDescr.0 sysUpTime.0 1.3.6.1.2.1.2.1.1

Name: sysDescr.0
Value: DG/UX TCP/IP SNMP AGENT

Name: sysUpTime.0
Value: 964922

Name: ifNumber.0
Value: 2
```

DIAGNOSTICS

Exit status is 0 upon success.

Exit status is −1 if there are errors parsing the command line.

Exit status is 1 if the agent returns an error.

SEE ALSO

getmany(1M), getnext(1M), setany(1M), snmpd(1M), trap_recv(1M), trap_send(1M), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

getty - set terminal type, modes, speed, and line discipline

SYNOPSIS

/usr/sbin/getty [-h] [-t *timeout*] *line* [*speed* [*type* [*linedisc*]]]

where:

timeout A login time limit in seconds.
line A TTY device file in the /dev directory.
speed The label of an entry in the /etc/ttydefs file.
type A value for the TERM variable (obsolete).
linedisc The initial line discipline (obsolete).

DESCRIPTION

getty is a symbolic link to /usr/lib/saf/ttymon. It is included for compatibility with previous releases for the few applications that still call getty directly. getty can be executed only by the superuser, that is, by a process with the user ID 0 (root).

Initially, getty generates a system identification message from the values returned by the uname(2) system call. Then, if /etc/issue exists, getty outputs the file's contents to the user's terminal, followed finally by the login prompt. getty then reads the user's login name and invokes the login(1) command with the user's name as its argument.

Initially, getty attempts to adapt the system to the terminal speed by using the options and arguments specified on the command line. Subsequently, as getty reads the user's name a character at a time, it attempts to customize the speed setting. If a null character (or framing error) is received, it is assumed to be the result of the user pressing the BREAK key. This will cause getty to attempt the next speed in the series. The series that getty tries is determined by the "hunt groups" it finds in /etc/ttydefs [see ttydefs(4)].

The only required argument is *line*, the name of a TTY line in /dev to which getty is to attach itself. getty uses this string as the name of a file in the /dev directory to open for reading and writing.

Options are:

-h specify that no hangup should occur. If the -h flag is **not** set, a hangup will be forced by setting the TTY port speed to zero before setting the speed to the default or specified speed.

-t *timeout*
specify that getty should exit if the open on the line succeeds and no one types a user name on the line within *timeout* seconds.

speed The optional second argument is a label to a speed and TTY definition in the file /etc/ttydefs. This definition tells getty at what speed to run initially, what the initial TTY settings are, and what speed to try next should the user indicate (by pressing the BREAK key) that the speed is inappropriate. The default *speed* is 300 baud.

type This argument is obsolete and will be ignored. (In releases of the DG/UX System prior to Revision 5.4, the optional third argument was a character string that was assigned to the TERM variable and exported.)

linedisc This argument is obsolete and will be ignored. (In releases of the DG/UX System prior to Revision 5.4, the optional fourth argument was a character string describing which line discipline to use in communicating with the

terminal.)

When given no optional arguments, `getty` specifies the following: the *speed* of the interface is set to 300 baud, any parity is allowed, new-line characters are converted to carriage return and line feed, and tab expansion is performed on the standard output.

FILES

`/etc/ttydefs`	TTY definitions file
`/etc/issue`	System identification message file

SEE ALSO

`ct`(1), `login`(1), `init`(1M), `sttydefs`(1M), `ttymon`(1M), `uname`(2), `inittab`(4), `ttydefs`(4), `termio`(7), `ttycompat`(7).

NOTE

The following "check option" invocation of `getty`, provided in releases of the DG/UX System prior to Revision 5.4, is obsolete and no longer supported:

```
/usr/sbin/getty -c file
```

Instead use the following command to list the contents of the `/etc/ttydefs` file and perform a validity check on the file:

```
sttydefs -l
```

NAME

gridman – menu interface for maintaining a High Availability Disk Array subsystem

SYNOPSIS

gridman

DESCRIPTION

Only the superuser can execute the gridman command.

You invoke the gridman program from the diskman program.

The gridman program contains a complete set of menu-driven procedures for setting up and maintaining a High Available Disk Array subsystem. This includes setting up arrays, mirrors or individual disks, removing physical units, installing microcode and monitoring subsystem performance.

For a complete explanation of the gridman program, see the *Operating the High Availability Disk Array Subsystem* manual.

SEE ALSO

diskman(1M), term(5).

NOTES

In order for the user interface to function correctly, the TERM setting in the user's environment must be properly set.

NAME

groupadd - add (create) a new group definition on the system

SYNOPSIS

groupadd [-g *gid* [-o]] *group*

where:

gid A non-negative decimal integer below MAXUID as defined in the <sys/param.h> header file.

group A string of printable characters that specifies the name of the new group. It may not include a colon (:) or newline (\n).

DESCRIPTION

The groupadd command creates a new group definition on the system by adding the appropriate entry to the /etc/group file.

The following options are available:

-g *gid* Set the group id for the new group. The group ID defaults to the next available (unique) number above the highest number currently assigned. For example, if groups 100, 105, and 200 are assigned as groups, the next default group number will be 201. (Group IDs from 0-99 are reserved.)

-o Allow the *gid* to be duplicated (non-unique).

FILES

/etc/group

DIAGNOSTICS

The groupadd command exits with one of the following values:

0 Success.

2 Invalid command syntax. A usage message for the groupadd command is displayed.

3 An invalid argument was provided to an option.

4 *gid* is not unique (when -o option is not used).

9 *group* is not unique.

10 Cannot update the /etc/group file.

SEE ALSO

groupdel(1M), groupmod(1M), listusers(1), logins(1M), useradd(1M), userdel(1M), usermod(1M).

NAME

`groupdel` - delete a group definition from the system

SYNOPSIS

`groupdel` *group*

where:

group A string of printable characters that specifies the group to be deleted

DESCRIPTION

The `groupdel` command deletes a group definition from the system. It deletes the appropriate entry from the `/etc/group` file.

FILES

`/etc/group`

DIAGNOSTICS

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

`0` Success.

`2` . Invalid command syntax. A usage message for the `groupdel` command is displayed.

`6` `group` does not exist.

`10` Cannot update the `/etc/group` file.

SEE ALSO

`groupadd`(1M), `groupmod`(1M), `listusers`(1), `logins`(1M), `useradd`(1M), `userdel`(1M), `usermod`(1M).

NAME

groupmod - modify a group definition on the system

SYNOPSIS

groupmod [-g *gid* [-o]] [-n *name*] group

where:

gid A non-negative decimal integer below MAXUID as defined in <sys/param.h>

name A string of printable characters specifying a new name for the group; it may not include a colon (:) or newline (\n)

group The current name of the group to be modified

DESCRIPTION

The groupmod command modifies the definition of the specified group by modifying the appropriate entry in the /etc/group file.

The following options are available:

-g *gid* Set the group id for the new group. The group ID defaults to the next available (unique) number above 99. (Group IDs from 0–99 are reserved.)

-o Allow the *gid* to be duplicated (non-unique).

-n *name*
Specify a new name for the group.

FILES

/etc/group

DIAGNOSTICS

The groupmod command exits with one of the following values:

0 Success.

2 Invalid command syntax. A usage message for the groupmod command is displayed.

3 An invalid argument was provided to an option.

4 *gid* is not unique (when the -o option is not used).

6 *group* does not exist.

9 *name* already exists as a group name.

10 Cannot update the /etc/group file.

SEE ALSO

groupadd(1M), groupdel(1M), listusers(1), logins(1M), useradd(1M), userdel(1M), usermod(1M).

NAME

`halt` - stop the system processor

SYNOPSIS

`/sbin/halt [ -lnqy ]`

DESCRIPTION

`Halt` writes out any information pending to the disks and then stops the processor.

`Halt` normally logs the system shutdown to the system log daemon, `syslogd`(1M), and places a shutdown record in the login accounting file `/etc/wtmp`. These actions are inhibited if the `-n` or `-q` options are present.

Options are:

`-l` Do not log the system shutdown to `syslogd`.

`-n` Prevent the *sync* before stopping.

`-q` Quick halt. No graceful shutdown is attempted.

`-y` Halt the system, even from a dialup terminal.

FILES

`/etc/wtmp` login accounting file

SEE ALSO

`init`(1M), `reboot`(1M), `shutdown`(1M), `syslogd`(1M).

NOTE

This command is equivalent to `init 0`.

NAME

helpadm - make changes to the help facility database

SYNOPSIS

/etc/helpadm

DESCRIPTION

The DG/UX system Help Facility Administration command, helpadm , allows DG/UX system administrators and command developers to define the content of the Help Facility database for specific commands and to monitor use of the Help Facility. The helpadm command can be executed only by login root, login bin, or a login that is a member of group bin.

The helpadm command prints a menu of 3 types of Help Facility data which can be modified, and 2 choices relating to monitoring use of the Help Facility. The five choices are:

- Modify *startup* data
- Add, modify, or delete a *glossary* term
- Add, modify, or delete command data (description, options, examples, and keywords)
- Prevent monitoring use of the Help Facility (login root and login bin only)
- Permit monitoring use of the Help Facility (login root and login bin only)

The user may make one of the above choices by entering its corresponding letter (given in the menu), or may exit to the shell by typing q (for "quit").

If one of the first three choices is chosen, then the user is prompted for additional information; specifically, which *startup* screen, *glossary* term definition, or command description is to be modified. The user may also be prompted for information to identify whether the changes to the database are additions, modifications, or deletions. If the user is modifying existing data or adding new data, then he is prompted to make the appropriate modifications/additions. If the user is deleting a *glossary* term or a command from the database, then he must respond affirmatively to the next query in order for the deletion to be done. In any case, before the user's changes are final, he must respond affirmatively when asked whether he is sure he wants his requested database changes to be done.

By default, helpadm will put the user into ed(1) to make additions/modifications to database information. If the user wishes to be put into a different editor, then he should set the environment variable EDITOR in his environment to the desired editor.

If the user chooses to monitor/prevent monitoring use of the Help Facility, the choice made is acted on with no further interaction by the user.

FILES

HELPLOG	/usr/lib/help/HELPLOG
helpclean	/usr/lib/help/helpclean

SEE ALSO

ed(1), glossary(1), help(1), locate(1), starter(1), usage(1).

NOTES

Operators of diskless clients will have to make sure that they have been given permission to write on the file system /usr/lib/help before using this command. If they do not have the correct permissions helpadm will fail whenever it tries create or modify a file in this file system.

When the DG/UX System is delivered to a customer, the login command sets the environment variable LOGNAME. If LOGNAME is not set, then the options to monitor/prevent monitoring use of the Help Facility may not work properly.

NAME

ifconfig - configure DG/UX System network interface

SYNOPSIS

ifconfig *interface* [*address* [netmask *mask*] [broadcast *b_addr*]] [metric *n*] [dstaddr *d_addr*] [start | stop]

where:

interface	A string that specifies the name and unit number of the network interface, such as inen0
address	A name found in the host database (/etc/hosts) or an Internet address expressed in the Internet standard dot notation
mask	A 32-bit number that identifies which bits of the host's Internet address indicate the subnet number
b_addr	An IP broadcast address
n	An integer greater than or equal to 0
d_addr	The address of the other end of a point-to-point connection

DESCRIPTION

The ifconfig command controls a network interface for the TCP/IP protocol stack. It assigns an address to a network interface, configures the network interface parameters, and stops and restarts message passing for that interface. You must use ifconfig when you bring an interface up to define its network address; you can also use it later to redefine an interface address.

If you omit the optional arguments, ifconfig displays the current configuration for the specified network interface.

Use the netmask option with address assignment to specify a network mask to use for subnetting. The broadcast option, which you also can use with address assignment, changes the IP broadcast address for the given interface to the specified value. You can change the interface address, the broadcast address, and the netmask mask only if the interface is stopped.

Routing protocols such as routed(1M) use the metric option to determine the relative cost of using a particular link.

The dstaddr argument specifies the address of the other end of a point-to-point connection.

The key words start | stop represent the following:

start:	Enables sending and receiving messages.
stop:	Disables sending and receiving messages.

If the interface is capable of broadcasting and the broadcast command line option is not supplied, ifconfig uses the default broadcast address for the interface. If the netmask command line option is not supplied, the default network mask for the address is used. The default will disable subnetting at the interface.

Only the superuser can change the configuration of a network interface.

EXAMPLES

```
ifconfig inen0 128.0.0.31
ifconfig inen0 hostB broadcast 128.0.0.0
ifconfig inen0 128.5.1.31 broadcast 128.5.1.0 netmask 0xffffff00
```

The first example assigns Internet address 128.0.0.31 to interface inen0 with the default broadcast address. The second example maps hostname hostB to an Internet address given in /etc/hosts and associates that address with interface inen0. It also sets the IP broadcast address to be 128.0.0.0. The third example assigns

the Internet address `128.5.1.31` to the interface `inen0`, sets the network mask to `0xffffff00` so that the high-order 24 bits of the address will be used as the Internet network number (network 128.5, subnet 1), and sets the `broadcast` address so that its host number part is all zeroes.

DIAGNOSTICS

The system displays messages when the specified interface does not exist, when the requested address is unknown, when the user invoking `ifconfig` is not the superuser, and when the `broadcast` value is not satisfactory. For example, the only acceptable `broadcast` values for unsubnetted class B addresses are as follows:

```
255.255.255.255
0.0.0.0
net-number.255.255
net-number.0.0
```

Though the first two broadcast values are valid, they specify to broadcast to all nodes in the Internet, so very few people would find them acceptable. To broadcast to a given network, specify the *net-number* (for example, `128.223`) in the network portion of the broadcast address, and either all 0's or all 1's (*255.255*) in the host portion. 0's are BSD 4.2 compatible; 1's are BSD 4.3 compatible.

Flags

All of the following flags should be present for a working interface:

`RUNNING`	LAN controller is working. It was activated either by the `netinit`(1M) command or by another protocol stack using the same LAN controller.
`STARTED`	Interface enabled for sending and receiving data. It is adjusted with `ifconfig start|stop`
`UP`	Interface is `STARTED` and `RUNNING`
`BROADCAST`	Interface has capability to broadcast (some interfaces, such as `loop`, do not support broadcasting)

SEE ALSO

`netinit`(1M), `routed`(1M).

NAME

inetd – Internet services server

SYNOPSIS

/usr/bin/inetd [-d] [*configuration-file*]

DESCRIPTION

The inetd process (daemon) listens for connections on the appropriate designated ports of the services specified in inetd.conf. When a connection is found, inetd invokes the server program specified by that configuration file for the service requested. Once a server is finished, inetd continues to listen on the socket (except in some cases which are described below).

Use the -d option to write various diagnostic messages to syslog at level LOG_NOTICE. See syslog(3C) for more information.

The inetd server itself provides a number of simple TCP-based services. These include echo, discard, chargen (character generator), daytime (human readable time), and time (machine readable time, in the form of the number of seconds since midnight, January 1, 1900). For details of these services, consult Request for Comments (RFC) 862, 863, 864, 867, and 868.

New services can be activated and existing services deleted or modified by editing the configuration file and then sending the inetd server a hangup signal, SIGHUP.

When you start inetd, the Internet server, it reads its configuration information from *configuration-file*. The default configuration file is /etc/inetd.conf. See inetd.conf(4M) for more information on the format of this file. There are two inetd.conf prototype files: inetd.conf_tcpip.proto and inetd.conf_nfs.proto. During setup, the inetd.conf file is established by concatenating these two prototype files.

FILES

/etc/inetd.conf
/etc/syslog.conf

SEE ALSO

syslog(3C), inetd.conf(4M), syslog.conf(5).

NAME

infocmp - compare or print out TERMINFO descriptions

SYNOPSIS

```
infocmp [-d] [-c] [-n] [-I] [-L] [-C] [-r] [-u]
[ -s  d|i|l|c ] [-v] [-V] [-1] [ -w width ]
[ -A directory ] [ -B directory ] [ termname ] ...
```

where:

width is the output width limit in columns
directory is the path to a TERMINFO directory hierarchy

DESCRIPTION

The infocmp command can be used to compare a binary terminfo(4) entry with other terminfo(4) entries, rewrite a terminfo(4) description to take advantage of the use= terminfo(4) field, or print out a terminfo(4) description from the binary file in a variety of formats. In all cases, the boolean fields will be printed first, followed by the numeric fields, followed by the string fields.

Default Options

If no options are specified and zero or one *termname*s (see term(5)) are specified, the -I option will be assumed, listing the terminfo(4) description. If more than one *termname* is specified, the -d option will be assumed, comparing the terminfo(4) descriptions.

Comparison Options [-d] [-c] [-n]

Infocmp compares the terminfo(4) description of the first terminal *termname* with each of the descriptions given by the entries for the other terminals' *termname*s. If a capability is defined for only one of the terminals, the value used for the other terminal being compared will depend on the type of the capability: F (false) for boolean variables, -1 for integer variables, and NULL for string variables.

-d produce a list of capabilities that are different between two entries. With this option, you can find out what two people did differently when creating separate entries for the same terminal, and analyze how two similar terminals differ from each other.

-c produce a list of capabilities that are common between the two entries. Capabilities that are not set are ignored. This option can be used as a quick check before using the -u option.

-n produce a list of capabilities that are in neither entry. If no *termname*s are given, the value of the environment variable TERM will be used for both of the *termname*s. This can be used as a quick check to see if anything was left out of the description.

Source Listing Options [-I] [-L] [-C] [-r]

The -I, -L, and -C options will produce a source listing for each terminal named.

-I use the terminfo(4) names in the listing

-L use the long C variable name listed in <term.h>

-C use the termcap(5) names

-r when using -C, put out all capabilities, not just standard termcap(5) variables.

If no *termname*s are given, the value of the environment variable TERM will be used for the terminal name.

The source produced by the -C option may be used directly as a termcap(5) entry, but not all of the parameterized strings may be changed to the termcap(5) format. Infocmp will attempt to convert most of the parameterized information, but that which it doesn't will be plainly marked in the output and commented out. These should be edited by hand.

All padding information for strings will be collected together and placed at the beginning of the string where termcap(5) expects it. Mandatory padding (padding information with a trailing '/') will become optional.

All termcap(5) variables no longer supported by terminfo(4), but which are derivable from other terminfo(4) variables, will be output. Not all terminfo(4) capabilities will be translated; only those variables which were part of termcap(5) will normally be output. Specifying the -r option will take off this restriction, allowing all capabilities to be output in termcap(5) form.

Note that it is not always possible to convert a terminfo(4) string capability into an equivalent termcap(5) format. This restriction exists because padding is collected to the beginning of the capability, not all capabilities are output, mandatory padding is not supported, and termcap(5) parameter sequences were not as flexible. Also, a subsequent conversion of the termcap(5) file back into terminfo(4) format will not necessarily reproduce the original terminfo(4) source.

Some common terminfo(4) parameter sequences, their termcap(5) equivalents, and some terminal types which commonly have such sequences, are:

terminfo(4)	termcap(5)	Representative Terminals
%p1%c	%.	adm, dg
%p1%d	%d	hp, ANSI standard, vt100
%p1%'x'%+%c	%+x	concept
%i	%i	ANSI standard, vt100
%p1%?%'x'%>%t%p1%'y'%+%;	%>xy	concept
%p2 is printed before %p1	%r	hp

Use= Option [-u]

-u produce a terminfo(4) source description of the first terminals' *termname* which is relative to the sum of the descriptions given by the entries for the other terminals' *termname*s. Infocmp does this by analyzing the differences between the first *termname* and the other *termname*s and producing a description with use= fields for the other terminals. In this manner, it is possible to retrofit generic terminfo(4) entries into a terminal's description. Or, if two similar terminal descriptions exist, but were coded at different times or by different people so that each is a full description, using infocmp will show what can be done to change one description to be relative to the other.

A capability will be printed, preceded by an at-sign (@) to delete it, if it does not exist for the first *termname*, but one of the other *termname* entries contains a value for it. A capability's value is printed if the value for the first *termname* is not found in any of the other *termname* entries, or if the first of the other *termname* entries that has this capability gives a different value for the capability than that in the first *termname*.

The order of the other *termname* entries is significant. Since the terminfo(4) compiler tic(1M) does a left-to-right scan of the capabilities, specifying two use= entries that contain differing values for the same capabilities will produce different results depending on the order in which the entries are given. Infocmp will flag any

such inconsistencies between the other *termname* entries as they are found.

Alternatively, specifying a capability *after* a `use=` entry that contains that capability will cause the second specification to be ignored. Using `infocmp` to recreate a description can be a useful check to make sure that everything was specified correctly in the original source description.

Another error that does not cause incorrect compiled files, but will slow down the compilation, is specifying extra `use=` fields that are unnecessary. `Infocmp` will flag any *termname* `use=` fields that were not needed.

Other Options [-s d|i|l|c] [-v] [-V] [-1] [-w *width*]

`-s` sort the fields within each type according to the argument below:

- `d` leave fields in the order that they are stored in the `terminfo`(4) database.
- `i` sort by `terminfo`(4) name.
- `l` sort by the long C variable name.
- `c` sort by the `termcap`(5) name.

If no `-s` option is given, the fields printed out will be sorted alphabetically by the `terminfo`(4) name within each type, except in the case of the `-C` or the `-L` options, which cause the sorting to be done by the `termcap`(5) name or the long C variable name, respectively.

`-v` print out tracing information on standard error as the program runs.

`-V` print out the version of the program in use and exit.

`-1` cause the fields to be printed out one to a line. Otherwise, the fields will be printed several to a line to a maximum width of 60 characters.

`-w` change the maximum line width to *width* characters.

Changing Databases [-A *directory*] [-B *directory*]

The location of the compiled `terminfo`(4) database is taken from the environment variable `TERMINFO`. If the variable is not defined, or the terminal is not found in that location, the system `terminfo`(4) database, in `/usr/share/lib/terminfo`, will be used. The options `-A` and `-B` may be used to override this location. The `-A` option will set `TERMINFO` for the first *termname* and the `-B` option will set `TERMINFO` for the other *termname*s. With this feature, it is possible to compare descriptions for a terminal with the same name located in two different databases. This is useful for comparing descriptions for the same terminal created by different people. Otherwise the terminals would have to be named differently in a single `terminfo`(4) database for a comparison to be made.

EXAMPLES

```
infocmp -L
```

Print out the `terminfo`(4) description of the default terminal, using the long name for each capability. The default terminal is the one specified by the environment variable `TERM`.

```
infocmp vt100 xterm
```

Compare the `terminfo`(4) descriptions of the VT100 terminal and the standard X Window System terminal emulator.

```
infocmp -C -A /usr/opt/terminfo kterm
```

Convert the terminfo(4) binary file /usr/opt/terminfo/k/kterm into a closely equivalent termcap(5) source description.

FILES

/usr/share/lib/terminfo/?/*
: compiled terminal description database

/usr/share/lib/termcap
: old, textual terminal description database

/usr/include/term.h
: terminfo(4) header file

DIAGNOSTICS

malloc is out of space!

There was not enough memory available to process all the terminal descriptions requested. Run infocmp several times, each time including a subset of the desired *termname*s.

use= order dependency found:

A value specified in one relative terminal specification was different from that in another relative terminal specification.

`use=*term*' did not add anything to the description.

A relative terminal specification *term* did not contribute anything to the final description.

Must have at least two terminal names for a comparison to be done.

The -u, -d, and -c options require at least two terminal names.

SEE ALSO

captoinfo(1M), tic(1M), curses(3X), terminfo(4), term(5), termcap(5).

NAME

`init, telinit` – process control initialization

SYNOPSIS

`/sbin/init [0i123456SsQqabc]`

`/sbin/telinit [0i123456SsQqabc]`

DESCRIPTION

`init` is a general process spawner. Its primary role is to create processes from information stored in the file `/etc/inittab` [see `inittab`(4)].

At any given time, the system is in one of eight possible run levels. A run level is a software configuration of the system under which only a selected group of processes exist. The processes spawned by `init` for each of these run levels is defined in `/etc/inittab`. `init` can be in one of eight run levels, `0-6` and `S` or `s` (run levels `S` and `s` are identical). The run level changes when a privileged user runs `/sbin/init`. This user-spawned `init` sends appropriate signals to the original `init` spawned by the operating system when the system was booted, telling it which run level to change to.

The following are the arguments to `init`.

`0` shut the machine down so it is safe to remove the power. Have the machine remove power if it can.

`i` put the system in installation mode. All local file systems are mounted and a small set of essential kernel processes are running. The `installman`(1M) program is invoked to perform initial installation steps.

`1` put the system in system administrator mode. All file systems are mounted. Only a small set of essential kernel processes are left running. This mode is for administrative tasks such as installing optional utility packages. All files are accessible and no users are logged in on the system.

`2` put the system in multi-user mode. All multi-user environment terminal processes and daemons are spawned. This state is commonly referred to as the multi-user state.

`3` start the remote file sharing processes and daemons. Mount and advertise remote resources. Run level `3` extends multi-user mode and is known as the remote-file-sharing state.

`4` is available to be defined as an alternative multi-user environment configuration. It is not necessary for system operation and is usually not used.

`5` Stop the system and go to the firmware monitor. Bringing the system to this state is functionally equivalent to bringing it to init state `s` then entering the `halt`(1M) command.

`6` Stop the system and reboot to the state defined by the `initdefault` entry in `/etc/inittab`. Bringing the system to this state is functionally equivalent to bringing it to init state `s` then entering the `reboot`(1M) command.

`a,b,c` process only those `/etc/inittab` entries having the `a`, `b`, or `c` run level set. These are pseudo-states, which may be defined to run certain commands, but which do not cause the current run level to

change.

Q,q re-examine /etc/inittab.

S,s enter single-user mode. When this occurs, the terminal which executed this command becomes the system console. This is the only run level that doesn't require the existence of a properly formatted /etc/inittab file. If this file does not exist, then by default the only legal run level that init can enter is the single-user mode. When the system comes up to S or s, file systems for users' files are not mounted and only essential kernel processes are running. When the system comes down to S or s, all mounted file systems remain mounted, and all processes started by init that should only be running in multi-user mode are killed. In addition, any process that has a utmp entry will be killed. This last condition insures that all port monitors started by the SAC are killed and all services started by these port monitors, including ttymon login services, are killed. Other processes not started directly by init will remain running. For example, cron remains running.

When a DG/UX system is booted, init is invoked and the following occurs. First, init attempts to fsck and mount /usr using the /usr entry in /etc/fstab. If there is no /etc/fstab but there is an /etc/fstab.proto, init copies /etc/fstab.proto to /etc/fstab and then attempts the mount operation. The sequence is equivalent to:

```
if    [ ! -f /etc/fstab ] &&
      [ -f /etc/fstab.proto ]
then
      cp /etc/fstab.proto /etc/fstab
fi
mount -f /
fsck -xq /usr
mount /usr
```

Next, init looks in /etc/inittab for the initdefault entry [see inittab(4)]. If there is one, init will usually use the run level specified in that entry as the initial run level to enter. If there is no initdefault entry in /etc/inittab, init requests that the user enter a run level from the virtual system console. If an S or s is entered, init goes to the single-user state. If /usr was not mounted successfully, then single-user state is entered regardless of the initdefault setting in inittab. In the single-user state the virtual console terminal is assigned to the user's terminal and is opened for reading and writing. The command /sbin/su is invoked and a message is generated on the physical console saying where the virtual console has been relocated. Use either init or telinit to signal init to change the run level of the system. Note that if the shell is terminated (via an end-of-file), init will only re-initialize to the single-user state if the /etc/inittab file does not exist.

If a 0 through 6 is entered, init enters the corresponding run level if /usr is mounted. init will not permit a state change if /usr is not mounted. Run levels 0, 5, and 6 are reserved states for shutting the system down. Run levels 2, 3, and 4 are available as multi-user operating states.

If this is the first time since power up that init has entered a run level other than single-user state, init first scans /etc/inittab for boot and bootwait entries [see inittab(4)]. These entries are performed before any other processing of /etc/inittab takes place, providing that the run level entered matches that of the

entry. In this way any special initialization of the operating system, such as mounting file systems, can take place before users are allowed onto the system. `init` then scans `/etc/inittab` and executes all other entries that are to be processed for that run level.

To spawn each process in `/etc/inittab`, `init` reads each entry and for each entry that should be respawned, it forks a child process. After it has spawned all of the processes specified by `/etc/inittab`, `init` waits for one of its descendant processes to die, a powerfail signal, or a signal from another `init` or `telinit` process to change the system's run level. When one of these conditions occurs, `init` re-examines `/etc/inittab`. New entries can be added to `/etc/inittab` at any time; however, `init` still waits for one of the above three conditions to occur before re-examining `/etc/inittab`. To get around this, `init Q` or `init q` command wakes `init` to re-examine `/etc/inittab` immediately.

When `init` comes up at boot time and whenever the system changes from the single-user state to another run state, `init` sets the `ioctl`(2) states of the virtual console to those modes saved in the file `/etc/ioctl.syscon`. This file is written by `init` whenever the single-user state is entered.

When a run level change request is made `init` sends the warning signal (`SIGTERM`) to all processes that are undefined in the target run level. `init` waits five seconds before forcibly terminating these processes via the kill signal (`SIGKILL`).

The shell running on each terminal will terminate when the user types an end-of-file or hangs up. When `init` receives a signal telling it that a process it spawned has died, it records the fact and the reason it died in `/etc/utmp` and `/etc/wtmp` if it exists [see `who`(1)]. A history of the processes spawned is kept in `/etc/wtmp`.

If `init` receives a `powerfail` signal (`SIGPWR`) it scans `/etc/inittab` for special entries of the type `powerfail` and `powerwait`. These entries are invoked (if the run levels permit) before any further processing takes place. In this way `init` can perform various cleanup and recording functions during the powerdown of the operating system.

`telinit`, which is linked to `/sbin/init`, is used to direct the actions of `init`. It takes a one-character argument and signals `init` to take the appropriate action.

FILES

`/etc/inittab`
`/etc/utmp`
`/etc/wtmp`
`/etc/ioctl.syscon`
`/dev/console`

SEE ALSO

`installman`(1M), `ttymon`(1M), `shutdown`(1M), `inittab`(4), `utmp`(4), `utmpx`(4), `termio`(7).
`login`(1), `sh`(1), `stty`(1), `who`(1) in the *User's Reference Manual.*
`kill`(2) in the *Programmer's Reference Manual.*

DIAGNOSTICS

If `init` finds that it is respawning an entry from `/etc/inittab` more than ten times in two minutes, it will assume that there is an error in the command string in the entry, and generate an error message on the system console. It will then refuse to respawn this entry until either five minutes has elapsed or it receives a signal from a user-spawned `init` or `telinit`. This prevents `init` from eating up system resources when someone makes a typographical error in the `inittab` file or a

program is removed that is referenced in `/etc/inittab`.

When attempting to boot the system, failure of `init` to prompt for a new run level may be because the virtual system console is linked to a device other than the physical system console.

NOTES

`init` and `telinit` can be run only by a privileged user.

The `S` or `s` state must not be used indiscriminately in the `/etc/inittab` file. A good rule to follow when modifying this file is to avoid adding this state to any line other than the `initdefault`.

If a default state is not specified in the `initdefault` entry in `/etc/inittab`, state "`s`" is entered.

If the `utmp` file cannot be created when booting the system, the system will boot to state "`s`" regardless of the state specified in the *initdefault* entry in `/etc/inittab`. This can happen if the `root` filesystem is not accessible.

NAME

initrarp – Initialize ARP table through Reverse Address Resolution Protocol

SYNOPSIS

/usr/bin/initrarp [-f *altdir*] [-a *altpath*]

DESCRIPTION

Use the initrarp command to initialize an OS server's ARP table. The Address Resolution Protocol (arp(6P)) and the Reverse Address Resolution Protocol (rarp(6P)) uses the ARP table to maintain Ethernet-to-Internet address translation information for all diskless clients. That information is requested by the server when you boot the diskless client.

The initrarp command searches the /tftpboot directory for diskless clients' second-stage bootstraps. The name of these files are hexadecimal representation of clients' Internet addresses. Characters after the first eight are ignored. For every entry in /tftpboot, initrarp finds the hostname that corresponds to the Internet address and then the Ethernet address of that host. This Ethernet-Internet address pair is then placed as a permanent entry in the server's ARP table using the arp command.

With the -f flag, initrarp uses the directory named *altdir* instead of /tftpboot to scan for second-stage bootstraps. The -a flag allows you to override the arp command used to set the address translation entries. By default, /usr/bin/arp is used. Usually, these flags should not be necessary.

The initrarp command is automatically invoked by start-up scripts when the system comes up to the init 3 level. The superuser can invoke it at any other time (when she or he adds an entry for yet another diskless workstation).

NOTE: The Internet and Ethernet address of every diskless client must be present in the /etc/hosts and /etc/ethers files or in the corresponding Network Information Service (NIS) maps.

SEE ALSO

arp(1M), hosts(4), ethers(4), arp(6P), rarp(6P).

NAME

`install` - install commands

SYNOPSIS

`/etc/install [ -c` *dira*`] [ -f` *dirb*`] [ -g` *group*`] [ -i ] [ -m` *mode*`] [ -n` *dirc*`] [ -o ] [ -s ] [ -u` *user*`]` *file* `[`*dirx* `...]`

DESCRIPTION

`Install` is a command most commonly used in makefiles (see `make`(1)) to install a *file* (updated target file) in a specific place within a file system. Each *file* is installed by copying it into the appropriate directory, thereby retaining the mode and owner of the original command. The program prints messages telling the user exactly which files it is replacing or creating and where they are going.

If you give no options or directories (*dirx ...*), *install* searches a set of default directories (`/bin`, `/usr/bin`, `/etc`, `/lib` , and `/usr/lib`, in that order) for a file with the same name as *file*. When the first occurrence is found, `install` issues a message saying that it is overwriting that file with *file*, and proceeds to do so. If the file is not found, the program states this and exits without further action.

If you specify one or more directories (*dirx ...*) after *file*, those directories will be searched before the directories specified in the default list.

Options are:

`-c` *dira* Install a new command (*file*) in the directory specified by *dira*, only if it is not found. If it is found, `install` issues a message saying that the file already exists, and exits without overwriting it. This option can be used alone or with the `-s` option.

`-f` *dirb* Install *file* in given directory, whether or not one already exists. If the file being installed does not already exist, the mode and owner of the new file are set to `755` and `bin`, respectively. If the file already exists, the mode and owner of the file stay the same. This option can be used alone or with the `-o` or `-s` options.

`-g` *group* Set the group of *file* to *group*. This option is available only to the superuser.

`-i` Ignore the default directory list, searching only through the given directories (*dirx ...*). This option can be used alone or with any other options other than `-c` and `-f`.

`-m` *mode* Set the mode of *file* to *mode*. This option is available only to the superuser.

`-n` *dirc* Put *file* in directory *dirc*, if the file is not found in any of the searched directories; set the mode and owner of the new file to `755` and `bin`, respectively. This option can be used alone or with any other options except `-c` and `-f`.

`-o` Save *file*, if found, by copying it to `OLD`*file* in the directory in which the file was found. This option is useful when installing a normally text busy file such as `/bin/sh` or `/etc/getty`, where the existing file cannot be removed. May be used alone or with any other options other than `-c`.

`-s` Suppress printing of messages other than error messages. This option can be used alone or with any other options.

`-u` *user* Set the owner of *file* to *user*. This option is available only to the superuser.

SEE ALSO

chmod(1), make(1).

NAME

`installf` – add a file to the software installation database

SYNOPSIS

`installf` [`-c` *class*] *pkginst pathname* [*ftype* [[*major minor*] [*mode owner group*]]

`installf` [`-c` *class*] *pkginst* -

`installf` `-f` [`-c` *class*] *pkginst*

DESCRIPTION

`Installf` informs the system that a pathname not listed in the `pkgmap` file is being created or modified. It should be invoked before any file modifications have occurred.

When the second synopsis is used, the pathname descriptions will be read from standard input. These descriptions are the same as would be given in the first synopsis but the information is given in the form of a list. (The descriptions should be in the form: *pathname* [*ftype* [[*major minor*] [*mode owner group*]].)

After all files have been appropriately created and/or modified, `installf` should be invoked with the `-f` synopsis to indicate that installation is final. Links will be created at this time and, if attribute information for a pathname was not specified during the original invocation of `installf` or was not already stored on the system, the current attribute values for the pathname will be stored. Otherwise, `installf` verifies that attribute values match those given on the command line, making corrections as necessary. In all cases, the current content information is calculated and stored appropriately.

`-c` *class* Class to which installed objects should be associated. Default class is `none`.

pkginst Name of package instance with which the pathname should be associated.

pathname Pathname that is being created or modified.

ftype A one-character field that indicates the file type. Possible file types include:

`f`	a standard executable or data file
`e`	a file to be edited upon installation or removal
`v`	volatile file (one whose contents are expected to change)
`d`	directory
`x`	an exclusive directory
`l`	linked file
`p`	named pipe
`c`	character special device
`b`	block special device
`s`	symbolic link

major The major device number. The field is only specified for block or character special devices.

minor The minor device number. The field is only specified for block or character special devices.

mode The octal mode of the file (for example, 0664). A question mark (`?`) indicates that the mode will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

owner The owner of the file (for example, `bin` or `root`). The field is limited to 14 characters in length. A question mark (`?`) indicates that the owner will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

`group` The group to which the file belongs (for example, `bin` or `sys`). The field is limited to 14 characters in length. A question mark (?) indicates that the group will be left unchanged, implying that the file already exists on the target machine. This field is not used for linked or symbolically linked files.

`-f` Indicates that installation is complete. This option is used with the final invocation of `installf` (for all files of a given class).

NOTES

When *ftype* is specified, all applicable fields, as shown below, must be defined:

ftype	*Required Fields*
p x d f v or e	mode owner group
c or b	major minor mode owner group

The `installf` command will create directories, named pipes and special devices on the original invocation. Links are created when `installf` is invoked with the `-f` option to indicate installation is complete.

Links should be specified as *path1=path2*. *path1* indicates the destination and *path2* indicates the source file.

Files installed with `installf` will be placed in the class *none*, unless a class is defined with the command. Subsequently, they will be removed when the associated package is deleted. If this file should not be deleted at the same time as the package, be certain to assign it to a class which is ignored at removal time. If special action is required for the file before removal, a class must be defined with the command and an appropriate class action script delivered with the package.

When classes are used, `installf` must be used as follows:

```
installf -c class1 ...
installf -f -c class1 ...
installf -c class2 ...
installf -f -c class2 ...
```

EXAMPLE

The following example shows the use of `installf` invoked from an optional preinstall or postinstall script:

```
#create /dev/xt directory
#(needs to be done before drvinstall)
installf $PKGINST /dev/xt d 755 root sys ||
            exit 2
majno=`/usr/sbin/drvinstall -m /etc/master.d/xt
     -d $BASEDIR/data/xt.o -v1.0` ||
            exit 2
i=00
while [ $i -lt $limit ]
do
    for j in 0 1 2 3 4 5 6 7
    do
```

```
            echo /dev/xt$i$j c $majno `expr $i * 8 + $j`
                 644 root sys |
            echo /dev/xt$i$j=/dev/xt/$i$j
        done
        i=`expr $i + 1`
        [ $i -le 9 ] && i="0$i" #add leading zero
    done | installf $PKGINST - || exit 2
    # finalized installation, create links
    installf -f $PKGINST || exit 2
```

SEE ALSO

pkgadd(1M), pkgask(1M), pkgchk(1), pkginfo(1), pkgmk(1), pkgparam(1), pkgproto(1), pkgtrans(1), pkgrm(1M), removef(1M).

NAME

`installman` - manage system installation

SYNOPSIS

`installman`

DESCRIPTION

The `installman` command manages system installation by presenting a series of steps which may be necessary for completing system installation. The steps include setting up packages and building and rebooting a custom kernel. Before each step, you are asked whether or not you would like to perform the step; if you answer `yes`, you are presented with queries for performing the step. The queries are the same as those presented by `sysadm`(1M).

This command is automatically run when the system is booted to run level `i` (installation mode). The steps are described below.

Setting up packages

Some software packages require certain setup functions to be performed before the software is operational. This step allows you to perform those setup functions for packages that have already been loaded on the system.

This step is equivalent to `sysadm`'s *Software* -> *Package* -> *Set up* operation.

Building a custom kernel

This step allows you to build a custom kernel for the system. A custom kernel lists the hardware devices attached to the system and lists pseudo-devices necessary for proper system operation.

This is equivalent to `sysadm`'s *System* -> *Kernel* -> *Auto Configure* operation. You may tune kernel parameters later using `sysadm`'s *System* -> *Kernel* -> *Build* operation.

Rebooting the system

This step allows you to reboot the system to use the custom kernel created in the previous step. If you do not perform this step, the system will continue to use the currently-running kernel until you reboot the system.

This operation is equivalent to `sysadm`'s *System* -> *Kernel* -> *Reboot* operation.

FILES

`/usr/lib/installman/C/installman.rc`
: `idl`(4) file describing the operations presented

DIAGNOSTICS

Exit status is always 0.

SEE ALSO

`idi`(1), `admkernel`(1M), `admpackage`(1M), `reboot`(1M), `sysadm`(1M), `idl`(4).

NAME

kbdcomp - compile att_kbd tables

SYNOPSIS

kbdcomp [-vrR] [-o *outfile*] [*infile*]

DESCRIPTION

kbdcomp compiles tables for use with the att_kbd *STREAMS* module, a programmable string-translation module. The module has two separate abilities, each of which may be used alone or in combination.

The first ability, *lookup*, is that of performing simple substitution of bytes in an input stream. This ability is based on a simple 256-entry lookup table (as there are 256 possible bit combinations for a byte). As input is received, each byte is looked up in the translation table, and the table value for that byte is substituted in place of the original byte. The process is quick, and can be performed on each *STREAMS* message with no message copying or duplication.

The second ability, *mapping* allows searching for occurrences of specified strings of bytes (or individual bytes) in an input stream, and substituting other strings (or bytes) for them as they are recognized. There are three kinds of mapping that are differentiated by the relationship between the number of bytes in the input and the number of bytes in the output. *One-many* mapping means that for a given byte in the input, *many* bytes are substituted. *Many-one* mapping means that for many bytes in the input *one* byte is substituted. *Many-many* mapping includes the other two types as a proper subset, but also includes substitution of *many* bytes in the input with *many* bytes of output. att_kbd can perform all three types of mapping. The *lookup* ability described in the previous paragraph (that is, what amounts to *one-one* mapping) is a common special case useful enough to be included separately. By using combinations of both *lookup* and *mapping*, a larger class of input translation and conversion problems can be solved than can be solved by the use of either alone.

During operation, processing occurs in two major passes: the lookup table pass *always* precedes string mapping. The string mapping procedure is non-recursive for a given table and there is no feedback mechanism (that is, input is scanned in order as received and output is not re-scanned for occurrences of recognizable input strings). As an example of mapping, suppose one wishes to translate all occurrences of the string `this` in an input stream into the string `there`. The module recognizes and buffers occurrences of the string `th` (as each byte is received); if the following character is `i`, it will also be buffered, but if `x` is then received, a mismatch is recognized and no translation occurs. Assuming `thi` has been buffered, if the next character seen is `s`, a match is recognized, the buffer containing `this` is discarded, and the string `there` replaces it.

It should be obvious that both input and output strings can be of any non-zero length (see however, the section below on limitations). Each string to be recognized and translated must be unique, and no complete input string may constitute the leading substring of any other (for example, one may not define `abc` and `ab` simultaneously, but may so define `abc`, `abd`, and `abxy`).

Given a filename (or standard input if no name is supplied), kbdcomp will compile tables into the output file specified by the -o option. If the -o option is not supplied, output is to the file **kbd.out**.

The -v option causes parsing and verification—no output file is produced; if no error messages are printed, then the input file is syntactically correct. The -r option causes the compiler to check for and report on byte values that cannot be generated

in a table (see the description below). The option `-R` is equivalent to `-r` but it tries to print printable characters as themselves rather than in octal format.

Input Language

Source files for `kbdcomp` are a series of table declarations. Within each table declaration are a number of definitions and functions. A table declaration is one of the forms `map`, `link`, or `extern`:

```
map type ( name ) { expressions }

link ( string )
extern ( string )
```

The `link` and `extern` forms will be described later below. The *name* of a `map` must be a *simple token* not containing any colons, commas, quotes, or spaces. (For our purposes, a *simple token* is a sequence of alphabetic and/or numeric characters with no embedded punctuation, white space, or special symbols.) The *type* field is an optional field that may be either of the keywords `full` or `sparse`. If omitted, the type defaults to `sparse`. The effect of this field is described in more detail below. The expressions contained in the `map` declaration are one of the following forms. Reserved keywords are printed in constant-width font, variables in italics:

```
keylist ( string string )
define ( word value )
word ( extension result )
string ( word word )
strlist ( string string )
error ( string )
timed
```

The `keylist` form is for defining lookup table entries while the remaining forms are the separate string functions.

The definition form (`define`) allows a mnemonic word (the first argument) to be associated with a string (the second argument). It is useful for replacing complicated sequences (for example, those containing special symbols or control characters) with mnemonic words to facilitate the design and readability of tables.

Using the *word* form (where *word* must be a previously defined sequence) in a manner similar to a `C` function call results in the *value* of *word* being concatenated with *extension*; when the combination is recognized at runtime, it is mapped to *result*. The *value* may be a string of characters or a single byte. The following is an illustration (not intended to be complete):

```
map (some_accents) {
     define(acute '\047')
     define(grave '`' )
     acute(a '\341')        # same as string("\047a" "\341")
     grave(a '\340')
     # ...et cetera...
     keylist("zyZY" "yzYZ")
}
```

This map (above) defines the single quote and reverse quote keys as *dead-keys*, which when followed by *a* produce a character from the *ISO 8859-1* codeset. It is not necessary for the definition, extension, or result to be a single byte; they may be arbitrary strings.

Strings in definitions and arguments may generally be entered either without quotation or between double quotes. Byte constants may likewise be entered unquoted or between single quotes. The only time quotation is strictly required is when the string contains parentheses, spaces, tab characters, or other special symbols. The language makes no real distinction between byte constants and string constants: both are treated as null-terminated strings; the choice of whether to use a one-character string or a byte constant is thus a matter of taste. Most quoting conventions of `C` are recognized, except that *octal* constants must be exactly three digits long. Octal constants may be used in strings as well. In the example above, the arguments to `keylist` need not be quoted, as they contain no special symbols. The following example illustrates some situations where strings *must* be quoted:

```
string(abc "two words")          # literal space
keylist("[{}]" "(())")           # brackets/parentheses
define(esc_seq "\033\t(") # tab and parenthesis
define(space ' ')                # literal space
string(abc "keylist")            # keyword used as argument
```

Comments in files (inside or outside of map declarations) may be entered in the same manner as for `sh`(1); that is, after a `#` at the end of a line, or on a line beginning with `#`, as shown in the above examples.

The `keylist` form allows single bytes to be mapped to other single bytes; it defines actions that are treated in the lookup table (that is, are performed before mapping). Any byte value that is not explicitly changed by being included in a `keylist` form will, of course, be left unchanged; if no `keylist` forms appear in a map definition, then `kbdcomp` does not generate a lookup table for the map, and the lookup phase is skipped during module operation. Each byte in the first string argument to `keylist` is mapped to the byte at *the same position* in the second string argument. That is, given two strings X and Y as arguments: X_i maps to Y_i, X_j maps to Y_j and so forth. The two arguments must, after evaluation, be found to contain the same number of bytes.

The `string` form has a function similar to mnemonic forms defined with `define` and may be used for any type of many-many mapping. The first argument to `string` is mapped to the second argument (see the comment in the sample map above).

Mappings using both `keylist` and `string` or any `define` forms may be combined: if *i* is mapped to *a* with a `keylist` form, and *a* is used in the sequence *`a*, then when the user types *`i*, the sequence *`a* is seen by the string mapping process (because *lookup* is done first) and translated accordingly.

The `keylist` form is intended mainly for use in simple keyboard re-arrangement and case-conversion applications; `string` is for one-many mapping or for isolated instances of many-many mapping; the `define` form and words defined with it are intended for more general use in groups of related sequences. In some situations while a one-one mapping with `keylist` may be an obvious choice, the same effect may be achieved with `string` forms to avoid having a contradictory mapping. For example, suppose one desires, simultaneously, to translate *x* into *y* and *y* into *abc*. If *x* is mapped to *y* via a `keylist` form and *y* is mapped to *abc* via a `string` form, then it may be impossible to obtain *y* itself (unless defined in another sequence), even though that was not the intention—the intention was to obtain *y* whenever the user enters *x*. This is a *contradictory mapping*:

```
keylist(x y)
string(y abc)        # "y" itself cannot be generated
```

There are cases where the intention is that *y* not be generated, but most often the intention is to generate it. This problem (a relatively common one in codeset mapping) can be "solved" by using a `string` form to map *x* to *y* initially rather than using a `keylist` form. This allows both *y* and *abc* to be generated:

```
string(x y)
string(y abc)
```

Entering a large number of one-one mappings with `string` can be somewhat tedious. To make things easier, the `strlist` form is provided. The two string arguments to `strlist` are interpreted in the same manner as arguments to `keylist`, (that is, they are one-one mappings) except that they are not done by the lookup table, but are processed as string mappings. In the following example, the first three `string` definitions can be reduced to the `strlist` form which follows:

```
string(a b)
string(c d)
string(e f)

strlist(ace bdf)
```

It is important to recognize the difference between `string` and `strlist`: with `string`, the two arguments are a *single* mapping definition (which may be of any type) whereas with `strlist`, one or more *one-one* string mappings are defined simultaneously. A set of mappings defined with a combination of `string` and `strlist` do *not* exhibit the same type of incompatibility described above between `keylist` and `string`.

Some further aspects of module processing can now be presented. When a partial match in an input sequence is detected during string processing, it is buffered; if at some point the match no longer succeeds, the *first byte* of the matched buffer is normally sent to the neighboring module. The rest of the input is left in the buffer and scanned again to see if it matches the beginning of another sequence. The `error` entry allows one to send a string (or byte) constant (called a *fallback character*) instead of the byte that began the previous sequence; this is particularly useful in codeset mapping and conversion applications where the character which failed to be translated might be one which does not occur or has some other meaning in the target codeset. The following (somewhat contrived) example illustrates use of the `error` form:

```
# turn arrow keys into vi commands
map (vi_map) {
     string("\033[A" k) # up
     string("\033[B" j) # down
     error("!")
}
```

Given input of the *escape* character followed by `[A` or `[B`, a single character (`j` or `k`) is generated. If presented with the sequence `escape-[Q`, the module will produce the sequence `![Q`. The error string `!` replaces *escape* because the sequence failed to match when `Q` was received. The remaining characters are re-scanned, and neither `[` nor `Q` is found to begin a recognized sequence.

One-one mapping with strings or other defined forms (rather than via a `keylist` lookup table) is generally performed with a linear search operation when looking for bytes which begin sequences. However, if the table is specified as a `full` table, it is initially indexed rather than searched linearly, and thus processed much more quickly

when there are a large number of entries. This should be kept in mind in codeset mapping applications where nearly all characters are mapped, and many (or most) are one-one mappings. If only a very few characters are mapped with string functions, one must decide on whether to trade a small gain in processing speed for the space needed to store the index if a table is made `full`.

The `link` form is used to produce a *composite table*. A composite table is really a form of linkage that allows several tables to be used together in sequence as if the sequence were a single table. The string argument to `link` is of the following form:

`composite:component`*1*`,component`*2*`,component`*n*

The target composite name is followed by a colon, and the ordered component list is comma-separated. If the string argument contains spaces or special characters, it must be quoted. (This string is *not* interpreted by `kbdcomp`, but is left intact in the output file; it is interpreted by the module at runtime.) When a composite table is used, the effect is similar to pushing more than one instance of the `att_kbd` module in the sense that the component tables function sequentially but it is accomplished within a single instance of the module. As output is produced by processing with one table in the composite, the data is subsequently processed by the next component and so forth until the final result emerges at the end of the sequence. (There is no restriction on the use of any combination of `full` and `sparse` tables in a composite.)

Composite tables are useful for simplifying complex mapping situations by modularizing the processing and for increasing the re-usability of tables for different mapping applications. Tables primarily implementing codeset mappings may be linked to other tables primarily implementing compose- or dead-key sequences. With a single table implementing a common codeset mapping, several different tables implementing combinations of codeset mapping and compose-key layouts may be built. A typical configuration might use one table for mapping from an external to internal codeset, then use one or more separate tables *working in the internal codeset* to provide compose- or dead-key functionality, as in the following example. One table, `646Sp-8859` maps from an ISO 646 variant (Spanish) external codeset to ISO 8859-1; this is combined with two other tables respectively implementing ISO 8859-1 by compose-sequences, and by dead-key sequences:

```
link("composed:646Sp-8859,8859-1-cmp")
link("deadkey:646Sp-8859,8859-1-dk")
```

Composite tables can also be built while the module is running from the `kbdload` command line [see `kbdload`(1M) for details]. The component tables are linked and processed in the given order (left-to-right). Because the `link` argument is actually parsed at run-time by the `att_kbd` module, it is not an error to refer to tables that are not contained in the file currently being compiled. An error will be generated when the file is loaded if any component of a link is not present in memory at that time.

The `extern` form can be used to declare an external function managed by the `alp` module. External functions are managed in a list by that module, and are available for use as if they were simple tables in `att_kbd`. External functions are not downloaded, they are resident in the kernel and merely accessed by the `att_kbd` module [see `alp`(7) for more information]. Such functions can also be declared dynamically when needed [see `kbdload`(1M)].

The directive `timed` may appear any place within a `map` declaration. If used, it causes the table within which it is defined to be interpreted in *timeout mode*. In this

mode, string mappings are considered to not match if more than a specified amount of time elapses after receipt of the first byte of a sequence without its being fully received and mapped. Given a timed map in which `abc` is to be mapped to `xyz` and the timeout value is 30, if the user types `ab`, then waits for longer than 30 time units before typing `c`, the entire sequence will not be translated. In this case the sequence is treated as any other mismatch would be: `a` is passed to the neighboring module, and `b` is checked to see if it begins a sequence. The timer is reset when a mismatch occurs, so that if `bc` is defined in this situation and `c` has just been received, it will be mapped as expected. The default timeout is typically 1/5 to 1/3 of a second [see `att_kbd`(7) for details].

Timeout mode is generally useful in situations where terminal *function keys* are being interpreted, to distinguish between a string typed by the user and a function key string sent by the terminal; it is not intended for use with "batch" applications such as the `iconv` command, nor generally in pipelines [see `pipe`(2)]. In a composite table, some components may be timed and some not, making the mode useful for combinations of codeset mapping and function key mapping.

Timing depends on several factors, including terminal baud-rate, system load, and the user's typing speed. If the timeout value is too long, then typed sequences that happen to be the same as function keys will be erroneously mapped; if the value is too short, then function keys may be missed under a heavy system load or with low speed devices. See `kbdset`(1) for information on how to change the timeout value, and `att_kbd`(7) for information on how an administrator may change the *default* timeout value. This directive should never be used in tables that implement codeset mapping, as it makes the results quite unpredictable. Long timeouts, on the order of seconds, may be useful in some contexts.

Building & Debugging

Users who intend to build their own tables may study the source tables supplied with the distribution in the directory `/usr/lib/kbd`.

If characters other than alpha-numerics are to be used, quoted strings are preferred to unquoted strings; quotation is required for some characters, as mentioned above. Map names and the first arguments of `define` should be alpha-numeric tokens.

The report generated by the `-r` option may be useful for debugging complex tables. The report (produced on standard error) consists of two octal lists. One list contains byte values that cannot be generated from the lookup table (if `keylist` forms are used). The other list contains byte values that cannot be generated in any way; in other words, values that are *neither* parts of "result text" (that is, products of string mappings) *nor* generated by the lookup table (if there is one), but that are *used* in other sequences. The report does not exhaustively list unreachable paths, but may indicate whether they exist and help pinpoint them.

Output Files

The files produced by `kbdcomp` begin with a header. The magic string is `kbd!map`, with a version number. This header is immediately followed by the tables themselves. (A file can contain more than one table.) The lines below can be added to the `/etc/magic` file for the `file` command to recognize `att_kbd` files.

```
0       string      kbd!map     att_kbd map file
>8      byte        >0          Ver %d:
>10     short       >0          with %d table(s)
```

LIMITATIONS

A maximum length of 128 bytes for input strings and 256 bytes for output strings is

imposed. The total amount of space consumed by a single table is limited to around 65,000 bytes. Versions are strictly incompatible; "object" tables are machine-dependent in their byte order and structure size. Thus, while source files are portable, the output of `kbdcomp` is not. This implies that when using remote devices across a network between heterogeneous machines, tables must be loaded on the machine where the module is actually pushed (that is, the remote side).

FILES

`/usr/lib/kbd` directory containing system standard map files

`/usr/lib/kbd/*.map`
source for some system map files

SEE ALSO

`kbdload`(1M), `kbdset`(1), `iconv`(1), `att_kbd`(7), `alp`(7), `cpz`(4M).

NAME

kbdload - load or link att_kbd tables

SYNOPSIS

kbdload [-p] *filename*

kbdload -u *table*

kbdload -l *string*

kbdload -L *string*

kbdload -e *string*

DESCRIPTION

Tables included in the file *filename* are *loaded* into the att_kbd *STREAMS* module, which must already have been pushed into the standard input *STREAM*. (In this context *loaded* means copied from a disk file into main memory within the operating system.) This program is intended both to provide for loading and linking of both *shared* or *public* tables and *private* tables implementing user-specific functionality. New users should refer to kbdcomp(1M) and att_kbd(7) for a general description of the module's capabilities.

Files are searched for only by the name given on the command line; no search path is implied. Tables loaded by the super user with the -p option from an absolute path beginning at /usr/lib/kbd are made publicly available and permanently resident, otherwise the loaded tables are available only to the caller, and are automatically unloaded when the att_kbd module is popped from the *STREAM*.

The -u option can be used to unload private tables and by the super-user to remove public tables. Tables may be unloaded only if they are not currently in use. (Tables which are members of *composite tables* always have non-zero reference counts since they are "used" in the composite; all composites which refer to them must be unloaded first.)

The -L and -l options are used for making composite tables on-the-fly. The -L option, if executed by the super-user, causes the composite to be made publicly available; it is otherwise private and -L is equivalent to -l. The string argument is constructed in the same manner as the link statement [see kbdcomp(1M)] in the compiler. If any component of the intended composite is not presently loaded in memory or if a component of a *public* table is not also *public*, an error message is printed and the linkage fails. More than one composite may be created in a single invocation by using either option sequentially.

The -e option with a string argument causes kbdload to declare to the att_kbd module a subroutine called *string*, which is assumed to be a subroutine managed by and registered with the alp module [see alp(7)]. These "external" subroutines may be used exactly as any other loaded table; they may participate as members of composite tables, etc.

Security Issues

Allowing users other than the super-user to load public tables is a security risk and is thus disallowed. (In general, any manipulation of a module instance by a user who is neither the super-user nor the user who originally pushed it is disallowed.) The library directory and all files contained in it should be protected by being *unwritable*. Administrators are encouraged to remember that the att_kbd system can be used to arbitrarily re-map the entire keyboard of a terminal, *as well as* the entire output *STREAM*; thus in extremely hostile environments, it might be prudent to remove execution permissions from kbdload for non-administrative users (for example, setting

the owner to *bin* or *root* and giving it a mode of 0500).

The `kbdload` command checks to insure that the real-uid of the invoker is the same as the *owner* of both standard input and standard output files, unless the real-uid of the invoking user is the super user. Paths to public tables are scrutinized for legitimacy. The `kbdload` command refuses to work as a *set-uid* program.

EXIT VALUES

Exit status is 0 if all tables could be loaded and/or all operations succeeded. In the event of any I/O error (for example, attempting to load a table with the same name as one already loaded and accessible to the caller) or failure to load a table, exit status is 1 and a message is printed indicating the error.

CAVEATS

Composite tables may be unloaded while they are actually in use without affecting current users, though *new* users may no longer attach to it. This is because composite tables are copied and expanded when they are attached in order to keep state information related to the attaching user. The "original" composite always has a zero reference count, and is never itself attached. This is not strictly a bug, it's an "anomaly"; the effect on the user is that a composite table may be attached and functional, yet not appear in the output of a `kbdset` query.

FILES

`/usr/lib/kbd` directory containing system standard map files

SEE ALSO

`kbdcomp`(1M), `kbdset`(1), `alp`(7), `att_kbd`(7).

NAME

keyserv - server for storing public and private keys

SYNOPSIS

keyserv [-n]

DESCRIPTION

NOTE: Secure RPC using DES Authentication is an additional feature that must be purchased separately from the DG/UX™ ONC™/NFS® package. You must have this feature to use the server described in this manual page.

Keyserv is a server (daemon) that is used for storing the private encryption keys of each user logged into the system. These encryption keys are used for accessing secure network services such as secure NFS. When a user logs in to the system, the login(1) program uses the login password to decrypt the user's encryption key stored in the Network Information Service, and then gives the decrypted key to keyserv to store away.

Normally, root's key is read from battery backed-up ram when keyserv starts up. This is useful during power-fail reboots when no one is around to type a password, yet you still want the secure network services to operate normally.

OPTIONS

-n Do not read root's key from battery backed-up ram. Instead, prompt the user for the password to decrypt root's key stored in the Network Information Service and then store the decrypted key in battery backed-up ram for future use. This option is useful if the key stored in battery backed-up ram ever gets out of date or corrupted.

SEE ALSO

login(1), publickey(4).

NAME

`killall` – kill all active processes

SYNOPSIS

`/etc/killall` [*signal* [*pid pid* ...]]

DESCRIPTION

`Killall` is a procedure used by `/etc/shutdown` to kill all active processes not directly related to the shutdown procedure. `Killall` terminates all processes with open files so that you can unmount the mounted file systems.

`Killall` sends *signal* (see `kill`(1)) to all remaining processes other than those described above. The default signal is 9. `Killall` will also accept a list of process id's to preserve. In this case, the first argument must be the signal number.

FILES

`/etc/shutdown`

SEE ALSO

`fuser`(1M), `kill`(1), `ps`(1), `shutdown`(1M), `signal`(2), `reboot`(2).

NAME

link, unlink – exercise link and unlink system calls

SYNOPSIS

/etc/link *file1 file2*
/etc/unlink *file*

DESCRIPTION

Link and unlink perform their respective system calls on their arguments, abandoning all error checking. These commands can be executed only by the super-user.

SEE ALSO

rm(1), link(2), unlink(2).

NOTE

This command will not link directories. However, it does not return an error message if you try to do so.

NAME

`listdgrp` – lists members of a device group

SYNOPSIS

`listdgrp` *dgroup* ...

DESCRIPTION

`listdgrp` displays the members of the device groups specified by the *dgroup* list.

DIAGNOSTICS

This command will exit with one of the following values:

0 successful completion of the task.

1 command syntax incorrect, invalid option used, or internal error occurred.

2 device group table could not be opened for reading.

3 device group *dgroup* could not be found in the device group table.

EXAMPLE

To list the devices that belong to group `partitions`:

```
$ listdgrp partitions
root
swap
usr
```

FILES

`/etc/dgroup.tab`

SEE ALSO

`getdgrp`(1M), `putdgrp`(1M).

NAME

`listen` - network listener server

SYNOPSIS

`/usr/lib/saf/listen` [`-m` *devstem*] *net_spec*

DESCRIPTION

The `listen` process "listens" to a network for service requests, accepts requests when they arrive, and invokes servers in response to those service requests. The network listener process may be used with any connection-oriented network (more precisely, with any connection-oriented transport provider) that conforms to the Transport Interface (TLI) specification.

The listener internally generates a pathname for the minor device for each connection; it is this pathname that is used in the `utmp` entry for a service, if one is created. By default, this pathname is the concatenation of the prefix `/dev/`*netspec* with the decimal representation of the minor device number. When the `-m` *devstem* option is specified, the listener will use *devstem* as the prefix for the pathname. In either case, the representation of the minor device number will be at least two digits (e.g., 05 or 27), but will be longer when necessary to accommodate minor device numbers larger than 99.

Server Invocation

When a connection indication is received, the listener creates a new transport endpoint and accepts the connection on that endpoint. Before giving the file descriptor for this new connection to the server, any designated STREAMS modules are pushed and the configuration script is executed, if one exists. This file descriptor is appropriate for use with either TLI (see especially `t_sync(3N)`) or the sockets interface library.

By default, a new instance of the server is invoked for each connection. When the server is invoked, file descriptor 0 refers to the transport endpoint, and is open for reading and writing. File descriptors 1 and 2 are copies of file descriptor 0; no other file descriptors are open. The service is invoked with the user and group IDs of the user name under which the service was registered with the listener, and with the current directory set to the HOME directory of that user.

Alternatively, a service may be registered so that the listener will pass connections to a standing server process through a FIFO or a named STREAM, instead of invoking the server anew for each connection. In this case, the connection is passed in the form of a file descriptor that refers to the new transport endpoint. Before the file descriptor is sent to the server, the listener interprets any configuration script registered for that service using `doconfig`(3N), although `doconfig` is invoked with both the NORUN and NOASSIGN flags. The server receives the file descriptor for the connection in a `strrecvfd` structure via an I_RECVFD `ioctl`(2).

For more details about the listener and its administration, see `nlsadmin`(1M).

FILES

`/etc/saf/`*pmtag*`/*`

SEE ALSO

`nlsadmin`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `doconfig`(3N), `nlsgetcall`, `nlsprovider`(3N), `streamio(7)`.
Network Programmer's Guide

NOTES

When passing a connection to a standing server, the user and group IDs contained in

the `strrecvfd` structure will be those for the listener (that is, they will both be 0); the user name under which the service was registered with the listener is not reflected in these IDs.

When operating multiple instances of the listener on a single transport provider, there is a potential race condition in the binding of addresses during initialization of the listeners if any of their services have dynamically assigned addresses. This condition would appear as an inability of the listener to bind a static-address service to its otherwise valid address, and would result from a dynamic-address service having been bound to that address by a different instance of the listener.

NAME

lockd – network lock server

SYNOPSIS

/usr/etc/rpc.lockd [-t *timeout*] [-g *graceperiod*]

DESCRIPTION

Lockd processes lock requests that are sent remotely by another lock server (daemon). Lockd forwards lock requests for remote data to the server site's lock server through the RPC/XDR(3N) package. Lockd then requests the status monitor server, statd(1M), for monitor service. The reply to the lock request will not be sent to the kernel until the status server and the server site's lock server have replied.

If either the status monitor or server site's lock server is unavailable, the reply to a lock request for remote data is delayed until all server programs become available.

When a server recovers, it waits for a grace period for all client site lockds to submit reclaim requests. Client site lockds, on the other hand, are notified by the statd of the server recovery and promptly resubmit previously granted lock requests. If a lockd fails to secure a previously granted lock at the server site, the lockd sends SIGLOST to a process.

OPTIONS

-t *timeout*

Lockd uses *timeout* (seconds) as the interval instead of the default value (15 seconds) to retransmit lock request to the remote server.

-g *graceperiod*

Lockd uses *graceperiod* (seconds) as the grace period duration instead of the default value (15 seconds).

SEE ALSO

statd(1M), fcntl(2), lockf(3C), signal(3C).

NAME

`logins` - list user and system login information

SYNOPSIS

`logins` [`-dmopstuxa`] [`-g` *groups*] [`-l` *logins*]

DESCRIPTION

This command displays information on user and system logins. Contents of the output is controlled by the command options and can include the following: user or system login, user id number, `/etc/passwd` account field value (user name or other information), primary group name, primary group id, multiple group names, multiple group ids, home directory, login shell, and four password aging parameters. The default information is the following: login id, user id, primary group name, primary group id and the account field value from `/etc/passwd`. Output is sorted by user id, displaying system logins followed by user logins.

`-d` Selects logins with duplicate uids.

`-m` Displays multiple group membership information.

`-o` Formats output into one line of colon-separated fields.

`-p` Selects logins with no passwords.

`-s` Selects all system logins.

`-t` Sorts output by login instead of by uid.

`-u` Selects all user logins.

`-x` Prints an extended set of information about each selected user. The extended information includes home directory, login shell and password aging information, each displayed on a separate line. The password information consists of password status (PS for passworded, NP for no password or LK for locked). If the login is passworded, status is followed by the date the password was last changed, the number of days required between changes, and the number of days allowed before a change is required. The password aging information shows the time interval that the user will receive a password expiration warning message (when logging on) before the password expires.

`-a` Adds two password expiration fields to the display. The fields show how many days a password can remain unused before it automatically becomes inactive and the date that the password will expire.

`-g` Selects all users belonging to `group`, sorted by login. Multiple groups can be specified as a comma-separated list.

`-l` Selects the requested login. Multiple logins can be specified as a comma-separated list.

FILES

`/etc/group`
`/etc/passwd`

SEE ALSO

groups(1), id(1), passwd(1), useradd(1M), usermod(1M), userdel(1M).

NOTES

Options may be used together. If so, any login matching any criteria will be displayed. When the `-l` and `-g` options are combined, a user will only be listed once, even if the user belongs to more than one of the selected groups.

NAME

lpadmin - configure the LP print service

SYNOPSIS

```
lpadmin -p printer options
lpadmin -x dest
lpadmin -d [dest]
lpadmin -S print-wheel -A alert-type [-W minutes] [-Q requests]
```

DESCRIPTION

lpadmin configures the LP print service by defining printers and devices. It is used to add and change printers, to remove printers from the service, to set or change the system default destination, to define alerts for printer faults, and to mount print wheels.

Adding or Changing a Printer

The first form of the lpadmin command (lpadmin -p *printer options*) is used to configure a new printer or to change the configuration of an existing printer. The following *options* may appear in any order.

-A *alert-type* [-W *minutes*]

The -A option is used to define an alert to inform the administrator when a printer fault is detected, and periodically thereafter, until the printer fault is cleared by the administrator. The *alert-types* are:

mail Send the alert message via mailx [see mailx(1)] to the administrator.

write Write the message to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is chosen arbitrarily.

quiet Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the fault has been cleared and printing resumes, messages will again be sent when another fault occurs with the printer.

none Do not send messages; any existing alert definition for the printer will be removed. No alert will be sent when the printer faults until a different alert-type (except quiet) is used.

shell-command

Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. Note that the mail and write values for this option are equivalent to the values mail *user-name* and write *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the su command to change to another user ID. If the su command has been used to change the user ID, then the *user-name* for the new ID is used.

list Display the type of the alert for the printer fault. No change is made to the alert.

The message sent appears as follows:

```
The printer printer has stopped printing for the reason given
below.  Fix the problem and bring the printer back on line.
```

```
Printing has stopped, but will be restarted in a few minutes;
issue an enable command if you want to restart sooner.  Unless
someone issues a change request

    lp -i request-id -P ...

to change the page list to print, the current request will be
reprinted from the beginning.

The reason(s) it stopped (multiple reasons indicate reprinted
attempts):

    reason
```

The LP print service can detect printer faults only through an adequate fast filter and only when the standard interface program or a suitable customized interface program is used. Furthermore, the level of recovery after a fault depends on the capabilities of the filter.

If the *printer* is `all`, the alerting defined in this command applies to all existing printers.

If the `-W` option is not used to arrange fault alerting for *printer*, the default procedure is to mail one message to the administrator of *printer* per fault. This is equivalent to specifying `-W once` or `-W 0`. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

`-c` *class*
Insert *printer* into the specified *class*. *Class* will be created if it does not already exist. (see `-r` to remove a *printer* from a *class*)

`-D` *comment*
Save this *comment* for display whenever a user asks for a full description of *printer* [see `lpstat`(1)]. The LP print service does not interpret this comment.

`-e` $printer_1$
Copy the interface program of an existing $printer_1$ to be the interface program for *printer*. (Options `-i` and `-m` may not be specified with this option.)

`-F` *fault-recovery*
This option specifies the recovery to be used for any print request that is stopped because of a printer fault, according to the value of *fault-recovery*:

`continue`
Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing.

`beginning`
Start printing the request again from the beginning.

`wait` Disable printing on *printer* and wait for the administrator or a user to enable printing again.

During the wait the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. (See the `-i` option of the `lp` command.) If no change request is made before printing is enabled, printing will resume at the top of the page where stopped, if the filter allows; otherwise, the request will be printed from the beginning.

`-f allow:`*form-list*
`-f deny:`*form-list*
Allow or deny the forms in *form-list* to be printed on *printer*. By default no forms are allowed on a new printer.

For each printer, the LP print service keeps two lists of forms: an "allow-list" of forms that may be used with the printer, and a "deny-list" of forms that may not be used with the printer. With the `-f allow` option, the forms listed are added to the allow-list and removed from the deny-list. With the `-f deny` option, the forms listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the forms in the list may be used on the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the forms in the deny-list may not be used with the printer. All forms can be excluded from a printer by specifying `-f deny:all`. All forms can be used on a printer (provided the printer can handle all the characteristics of each form) by specifying `-f allow:all`.

The LP print service uses this information as a set of guidelines for determining where a form can be mounted. Administrators, however, are not restricted from mounting a form on any printer. If mounting a form on a particular printer is in disagreement with the information in the allow-list or deny-list, the administrator is warned but the mount is accepted. Nonetheless, if a user attempts to issue a print or change request for a form and printer combination that is in disagreement with the information, the request is accepted only if the form is currently mounted on the printer. If the form is later unmounted before the request can print, the request is canceled and the user is notified by mail.

If the administrator tries to specify a form as acceptable for use on a printer that doesn't have the capabilities needed by the form, the command is rejected.

Note the other use of `-f`, with the `-M` option, below.

`-h` Indicate that the device associated with the printer is hardwired. If neither of the mutually exclusive options, `-h` and `-l`, is specified, this option is assumed.

`-I` *content-type-list*
Allow *printer* to handle print requests with the content types listed in a *content-type-list*. If the list includes names of more than one type, the names must be separated by commas or blank spaces. (If they are separated by blank spaces, the entire list must be enclosed in double quotes.)

The type `simple` is recognized as the default content type for files in the UNIX system. A `simple` type of file is a data stream containing only printable ASCII characters and the following control characters.

Control Character	Octal Value	Meaning
backspace	10_8	move back one character, except at beginning of line
tab	11_8	move to next tab stop
linefeed (newline)	12_8	move to beginning of next line
form feed	14_8	move to beginning of next page
carriage return	15_8	move to beginning of current line

To prevent the print service from considering `simple` a valid type for the printer, specify either an explicit value (such as the printer type) in the *content-type-list*, or an empty list. If you do want `simple` included along with other

types, you must include `simple` in the *content-type-list*.

Except for `simple`, each *content-type* name is freely determined by the administrator. If the printer type is specified by the `-T` option, then the printer type is implicitly considered to be also a valid content type.

`-i` *interface*
Establish a new interface program for *printer*. *Interface* is the pathname of the new program. (The `-e` and `-m` options may not be specified with this option.)

`-l` Indicate that the device associated with *printer* is a login terminal. The LP scheduler (`lpsched`) disables all login terminals automatically each time it is started. (The `-h` option may not be specified with this option.)

`-M -f` *form-name* [`-a` [`-o filebreak`]]
Mount the form *form-name* on *printer*. Print requests that need the pre-printed form *form-name* will be printed on *printer*. If more than one printer has the form mounted and the user has specified `any` (with the `-d` option of the `lp` command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

The page length and width, and character and line pitches needed by the form are compared with those allowed for the printer, by checking the capabilities in the `terminfo` database for the type of printer. If the form requires attributes that are not available with the printer, the administrator is warned but the mount is accepted. If the form lists a print wheel as mandatory, but the print wheel mounted on the printer is different, the administrator is also warned but the mount is accepted.

If the `-a` option is given, an alignment pattern is printed, preceded by the same initialization of the physical printer that precedes a normal print request, with one exception: no banner page is printed. Printing is assumed to start at the top of the first page of the form. After the pattern is printed, the administrator can adjust the mounted form in the printer and press return for another alignment pattern (no initialization this time), and can continue printing as many alignment patterns as desired. The administrator can quit the printing of alignment patterns by typing `q`.

If the `-o filebreak` option is given, a formfeed is inserted between each copy of the alignment pattern. By default, the alignment pattern is assumed to correctly fill a form, so no formfeed is added.

A form is "unmounted" either by mounting a new form in its place or by using the `-f none` option. By default, a new printer has no form mounted.

Note the other use of `-f` without the `-M` option above.

`-M -S` *print-wheel*
Mount the *print-wheel* on *printer*. Print requests that need the *print-wheel* will be printed on *printer*. If more than one printer has *print-wheel* mounted and the user has specified `any` (with the `-d` option of the `lp` command) as the printer destination, then the print request will be printed on the one printer that also meets the other needs of the request.

If the *print-wheel* is not listed as acceptable for the printer, the administrator is warned but the mount is accepted. If the printer does not take print wheels, the command is rejected.

A print wheel is "unmounted" either by mounting a new print wheel in its place or by using the option `-S none`. By default, a new printer has no print wheel

mounted.

Note the other uses of the `-S` option without the `-M` option described below.

`-m` *model*
Select *model* interface program, provided with the LP print service, for the printer. (Options `-e` and `-i` may not be specified with this option.)

`-o` *printing-option*
Each `-o` option in the list below is the default given to an interface program if the option is not taken from a preprinted form description or is not explicitly given by the user submitting a request [see `lp`(1)]. The only `-o` options that can have defaults defined are listed below.

`length=`*scaled-decimal-number*
`width=`*scaled-decimal-number*
`cpi=`*scaled-decimal-number*
`lpi=`*scaled-decimal-number*
`stty=`*'stty-option-list'*

The term "scaled-decimal-number" refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a "trailing" letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing `c`); numbers that show sizes in inches (marked with a trailing `i`); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch, or characters per inch.

The first four default option values must agree with the capabilities of the type of physical printer, as defined in the `terminfo` database for the printer type. If they do not, the command is rejected.

The *stty-option-list* is not checked for allowed values, but is passed directly to the `stty` program by the standard interface program. Any error messages produced by `stty` when a request is processed (by the standard interface program) are mailed to the user submitting the request.

For each printing option not specified, the defaults for the following attributes are defined in the `terminfo` entry for the specified printer type.

```
length
width
cpi
lpi
```

The default for `stty` is

```
stty='9600 cs8 -cstopb -parenb ixon
    -ixany opost -olcuc onlcr -ocrnl -onocr
    -onlret -ofill nl0 cr0 tab0 bs0 vt0 ff0'
```

You can set any of the `-o` options to the default values (which vary for different types of printers), by typing them without assigned values, as follows:

```
length=
width=
cpi=
lpi=
stty=
```

`-o nobanner`
Allow a user to submit a print request specifying that no banner page be printed.

`-o banner`
Force a banner page to be printed with every print request, even when a user asks for no banner page. This is the default; you must specify `-o nobanner` if you want to allow users to be able to specify `-o nobanner` with the `lp` command.

`-r` *class*
Remove *printer* from the specified *class*. If *printer* is the last member of *class*, then *class* will be removed. (see `-c` to add *printers* to a *class*)

`-S` *list*
Allow either the print wheels or aliases for character sets named in *list* to be used on the printer.

If the printer is a type that takes print wheels, then *list* is a comma or space separated list of print wheel names. (Enclose the list with quotes if it contains blanks.) These will be the only print wheels considered mountable on the printer. (You can always force a different print wheel to be mounted, however.) Until the option is used to specify a list, no print wheels will be considered mountable on the printer, and print requests that ask for a particular print wheel with this printer will be rejected.

If the printer is a type that has selectable character sets, then *list* is a comma or blank separated list of character set name "mappings" or aliases. (Enclose the list with quotes if it contains blanks.) Each "mapping" is of the form

known-name=alias

The *known-name* is a character set number preceded by `cs` (such as `cs3` for character set three) or a character set name from the `Terminfo` database entry `csnm`. [See `terminfo`(4) in the *Programmer's Reference Manual*.] If this option is not used to specify a list, only the names already known from the Terminfo database or numbers with a prefix of `cs` will be acceptable for the printer.

If *list* is the word `none`, any existing print wheel lists or character set aliases will be removed.

Note the other uses of the `-S` with the `-M` option described above.

`-s` *system-name*[*!printer-name*]
Make a remote printer (one that must be accessed through another system) accessible to users on your system. *System-name* is the name of the remote system running DG/UX 5.4x on which the remote printer is located; it must be listed in the systems table (`/etc/lp/Systems`). *Printer-name* is the name used on the remote system for that printer. See the `-U` option for accessing printers on systems running pre-DG/UX 5.4x. For example, if you want to access $printer_1$ on $system_1$ and you want it called $printer_2$ on your system, enter `-p` $printer_2$ `-s` $system_1$`!`$printer_1$

`-T` *printer-type-list*
Identify the printer as being of one or more *printer-types*. Each *printer-type* is used to extract data from the `terminfo` database; this information is used to initialize the printer before printing each user's request. Some filters may also

use a *printer-type* to convert content for the printer. If this option is not used, the default *printer-type* will be `unknown`; no information will be extracted from `terminfo` so each user request will be printed without first initializing the printer. Also, this option must be used if the following are to work: `-o cpi`, `-o lpi`, `-o width`, and `-o length` options of the `lpadmin` and `lp` commands, and the `-S` and `-f` options of the `lpadmin` command.

If the *printer-type-list* contains more than one type, then the *content-type-list* of the `-I` option must either be specified as `simple`, as empty (`-I ""`), or not specified at all.

`-u allow:`*login-ID-list*
`-u deny:`*login-ID-list*
Allow or deny the users in *login-ID-list* access to the printer. By default all users are allowed on a new printer. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	a user on any system
system-name`!`*login-ID*	a user on system *system-name*
system-name`!all`	all users on system *system-name*
`all!`*login-ID*	a user on all systems
`all`	all users on all systems

For each printer the LP print service keeps two lists of users: an "allow-list" of people allowed to use the printer, and a "deny-list" of people denied access to the printer. With the `-u allow` option, the users listed are added to the allow-list and removed from the deny-list. With the `-u deny` option, the users listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the users in the list may use the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the users in the deny-list may not use the printer. All users can be denied access to the printer by specifying `-u deny:all`. All users may use the printer by specifying `-u allow:all`.

`-U` *dial-info*
The `-U` option allows your print service to access a remote printer. (It does not enable your print service to access a remote printer service.) Specifically, `-U` assigns the "dialing" information *dial-info* to the printer. *Dial-info* is used with the `dial` routine to call the printer. Any network connection supported by the Basic Networking Utilities will work. *Dial-info* can be either a phone number for a modem connection, or a system name for other kinds of connections. Or, if `-U direct` is given, no dialing will take place, because the name `direct` is reserved for a printer that is directly connected. If a system name is given, it is used to search for connection details from the file `/etc/uucp/Systems` or related files. The Basic Networking Utilities are required to support this option. By default, `-U direct` is assumed.

`-U` *printer-name@system-name*
The `-U` option may also be used to make available a printer that runs on a pre-DG/UX 5.4x system. This is similar to the `-s` *system-name* option which makes printers available on remote systems that are DG/UX 5.4x systems.

`-v` *device*
Associate a *device* with *printer*. *Device* is the path name of a file that is writable

by `lp`. Note that the same *device* can be associated with more than one printer.

Restrictions

When creating a new printer, one of three options (`-v`, `-U`, or `-s`) must be supplied. In addition, only one of the following may be supplied: `-e`, `-i`, or `-m`; if none of these three options is supplied, the model standard is used. The `-h` and `-l` options are mutually exclusive. Printer and class names may be no longer than 14 characters and must consist entirely of the characters `A-Z`, `a-z`, `0-9` and _ (underscore). If `-s` is specified, the following options are invalid: `-A`, `-e`, `-F`, `-h`, `-i`, `-l`, `-M`, `-m`, `-o`, `-U`, `-v`, and `-W`.

Removing a Printer Destination

The `-x` *dest* option removes the destination *dest* (a printer or a class), from the LP print service. If *dest* is a printer and is the only member of a class, then the class will be deleted, too. If *dest* is `all`, all printers and classes are removed. No other *options* are allowed with `-x`.

Setting/Changing the System Default Destination

The `-d` [*dest*] option makes *dest*, an existing printer or class, the new system default destination. If *dest* is not supplied, then there is no system default destination. No other *options* are allowed with `-d`.

Setting an Alert for a Print Wheel

`-S` *print-wheel* `-A` *alert-type* [`-W` *minutes*] [`-Q` *requests*]

The `-S` *print-wheel* option is used with the `-A` *alert-type* option to define an alert to mount the print wheel when there are jobs queued for it. If this command is not used to arrange alerting for a print wheel, no alert will be sent for the print wheel. Note the other use of `-A`, with the `-p` option, above.

The *alert-types* are:

`mail` Send the alert message via the `mailx` command to the administrator.

`write` Write the message, via the `write` command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.

`quiet` Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the *print-wheel* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the `-Q` option.

`none` Do not send messages until the `-A` option is given again with a different *alert-type* (other than `quiet`).

shell-command

Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. Note that the `mail` and `write` values for this option are equivalent to the values `mail` *user-name* and `write` *user-name* respectively, where *user-name* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another user ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list` Display the type of the alert for the print wheel on standard output. No change is made to the alert.

The message sent appears as follows:

```
The print wheel print-wheel needs to be mounted
on the printer(s):
printer (integer1 requests)
integer2 print requests await this print wheel.
```

The printers listed are those that the administrator had earlier specified were candidates for this print wheel. The number *integer*$_1$ listed next to each printer is the number of requests eligible for the printer. The number *integer*$_2$ shown after the printer list is the total number of requests awaiting the print wheel. It will be less than the sum of the other numbers if some requests can be handled by more than one printer.

If the *print-wheel* is `all`, the alerting defined in this command applies to all print wheels already defined to have an alert.

If the `-W` option is not given, the default procedure is that only one message will be sent per need to mount the print wheel. Not specifying the `-W` option is equivalent to specifying `-W once` or `-W 0`. If *minutes* is a number greater than zero, an alert will be sent at intervals specified by *minutes*.

If the `-Q` option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the print wheel are waiting. If the `-Q` option is not given, or *requests* is 1 or the word `any` (which are both the default), a message is sent as soon as anyone submits a print request for the print wheel when it is not mounted.

FILES

`/var/spool/lp/*`
`/etc/lp`

SEE ALSO

`accept`(1M), `lpsched`(1M), and `lpsystem`(1M).
`enable`(1), `lp`(1), `lpstat`(1), and `stty`(1) in the *User's Reference for the DG/UX System*.
`dial`(3C), `terminfo`(4) in the *Programmer's Reference for the DG/UX System*.

NAME

lpc – line printer control program

SYNOPSIS

/usr/etc/lpc [*command* [*argument* ...]]

DESCRIPTION

Lpc is used by the system administrator to control the operation of the line printer system. For each line printer configured in /etc/printcap, lpc may be used to:

- disable or enable a printer,
- disable or enable a printer's spooling queue,
- rearrange the order of jobs in a spooling queue,
- find the status of printers, and their associated spooling queues and printer dameons.

Without any arguments, lpc will prompt for commands from the standard input. If arguments are supplied, lpc interprets the first argument as a command and the remaining arguments as parameters to the command. The standard input may be redirected causing lpc to read commands from file. Commands may be abbreviated; following is the list of recognized commands.

? [*command* ...]

help [*command* ...]
: Print a short description of each command specified in the argument list, or, if no arguments are given, a list of the recognized commands.

abort { all | *printer* ... }
: Terminate an active spooling daemon on the local host immediately and then disable printing (preventing new daemons from being started by lpr) for the specified printers.

clean { all | *printer* ... }
: Remove any temporary files, data files, and control files that cannot be printed (i.e., do not form a complete printer job) from the specified printer queue(s) on the local machine.

disable { all | *printer* ... }
: Turn the specified printer queues off. This prevents new printer jobs from being entered into the queue by lpr.

down { all | *printer* } *message* ...
: Turn the specified printer queue off, disable printing and put *message* in the printer status file. The message doesn't need to be quoted, the remaining arguments are treated like echo(1). This is normally used to take a printer down and let others know why (lpq will indicate the printer is down and print the status message).

enable { all | *printer* ... }
: Enable spooling on the local queue for the listed printers. This will allow lpr to put new jobs in the spool queue.

exit

quit
: Exit from lpc.

restart { all | *printer* ... }
: Attempt to start a new printer daemon. This is useful when some abnormal

condition causes the daemon to die unexpectedly leaving jobs in the queue. `Lpq` will report that there is no daemon present when this condition occurs. If the user is the super-user, try to abort the current daemon first (i.e., kill and restart a stuck daemon).

`start` { `all` | *printer* ... }
Enable printing and start a spooling daemon for the listed printers.

`status` { `all` | *printer* ... }
Display the status of daemons and queues on the local machine.

`stop` { `all` | *printer* ... }
Stop a spooling daemon after the current job completes and disable printing.

`topq` *printer* [*jobnum* ...] [*user* ...]
Place the jobs in the order listed at the top of the printer queue.

`up` { `all` | *printer* ... }
Enable everything and start a new printer daemon. Undoes the effects of *down*.

FILES

`/etc/printcap`	printer description file
`/usr/spool/*`	spool directories
`/usr/spool/*/lock`	lock file for queue control

DIAGNOSTICS

?Ambiguous command	abbreviation matches more than one command
?Invalid command	no match was found
?Privileged command	command can be executed by root only

SEE ALSO

`lpr`(1), `lpq`(1), `lprm`(1) in the *User's Reference*; `lpd`(1M); `printcap`(5) in the *Programmer's Reference*.

NAME

lpd – line printer spooler

SYNOPSIS

/usr/lib/lpd [-l] [*port#*]

DESCRIPTION

Lpd is the line printer spool area handler (daemon) and is normally invoked at boot time from the *rc.lpsched* file if the system administrator has set the *lpd_START* variable in */etc/dgux.params* to *true*. The system administrator must also set up printers in */etc/printcap*. and the corresponding spooling areas. Lpd then makes a single pass through this printcap(5) file to find out about the existing printers and prints any files left after a crash. It then uses the system calls listen(2) and accept(2) to receive requests to print files in the queue, transfer files to the spooling area, display the queue, or remove jobs from the queue. In each case, it forks a child to handle the request so the parent can continue to listen for more requests. The Internet port number used to rendezvous with other processes is normally obtained with getservbyname(3N) but can be changed with the *port#* argument. The -l flag causes lpd to log valid requests received from the network. This can be useful for debugging purposes.

Note that by default the AT&T lpsched(1M) BSD emulation is listening to the BSD printer network port. If you desire to use the BSD lpd, you must delete the *Port Services* entry for this emulation mode. This can be accomplished through sysadm(1M) with the sequence Device -> Port -> Port Service -> Delete -> tcp -> lpd. The original settings can be found in */etc/saf/tcp/_pmtab.proto.*

Access control is provided by two means. First, All requests must come from one of the machines listed in the file */etc/hosts.equiv* or */etc/hosts.lpd*. Second, if the "rs" capability is specified in the *printcap* entry for the printer being accessed, lpr requests will only be honored for those users with accounts on the machine with the printer. Finally the printer system maintains protected spooling areas so that users cannot access queued files, but the printer processes can. Thus the spooling areas setup by the system administrator *must* have mode *660* with *spooler user* and *spooler group.*

The file *minfree* in each spool directory contains the number of disk blocks to leave free so that the line printer queue won't completely fill the disk. The *minfree* file can be edited with your favorite text editor.

The file *lock* in each spool directory is used to prevent multiple spoolers from becoming active simultaneously, and to store information about the spooler process for lpr(1), lpq(1), and lprm(1). After the spooler has successfully set the lock, it scans the directory for files beginning with *cf*. Lines in each *cf* file specify files to be printed or non-printing actions to be performed. Each such line begins with a key character to specify what to do with the remainder of the line.

J Job Name. String to be used for the job name on the burst page.

C Classification. String to be used for the classification line on the burst page.

L Literal. The line contains identification info from the password file and causes the banner page to be printed.

T Title. String to be used as the title for pr(1).

H Host Name. Name of the machine where lpr was invoked.

P Person. Login name of the person who invoked lpr. This is used to verify ownership by lprm.

M	Send mail to the specified user when the current print job completes.
f	Formatted File. Name of a file to print which is already formatted.
l	Like "f" but passes control characters and does not make page breaks.
p	Name of a file to print using `pr`(1) as a filter.
t	Troff File. The file contains `troff`(1) output (cat phototypesetter commands).
n	Ditroff File. The file contains device independent troff output.
d	DVI File. The file contains *Tex*(l) output (DVI format from Standford).
g	Graph File. The file contains data produced by `plot`(3X).
c	Cifplot File. The file contains data produced by *cifplot*.
v	The file contains a raster image.
r	The file contains text data with FORTRAN carriage control characters.
1	Troff Font R. Name of the font file to use instead of the default.
2	Troff Font I. Name of the font file to use instead of the default.
3	Troff Font B. Name of the font file to use instead of the default.
4	Troff Font S. Name of the font file to use instead of the default.
W	Width. Changes the page width (in characters) used by `pr`(1) and the text filters.
I	Indent. The number of characters to indent the output by (in ascii).
U	Unlink. Name of file to remove upon completion of printing.
N	File name. The name of the file which is being printed, or a blank for the standard input (when `lpr` is invoked in a pipeline).

If a file can not be opened, a message will be logged via `syslog`(3C) using the *LOG_LPR* facility. *Lpd* will try up to 20 times to reopen a file it expects to be there, after which it will skip the file to be printed.

Lpd uses `dg_flock`(2) to provide exclusive access to the lock file and to prevent multiple spoolers from becoming active simultaneously. If the spooler hould be killed or die unexpectedly, the lock file need not be removed. The lock file is kept in a readable ASCII form and contains two lines. The first is the process id of the spooler and the second is the control file name of the current job being printed. The second line is updated to reflect the current status of *lpd* for the programs `lpq`(1) and `lprm`(1).

FILES

`/etc/printcap`	printer description file
`/usr/spool/*`	spool directories
`/usr/spool/*/minfree`	minimum free space to leave
`/dev/lp*`	line printer devices
`/dev/printer`	socket for local requests
`/etc/hosts.equiv`	lists machine names allowed printer access
`/etc/hosts.lpd`	lists machine names allowed printer access, but not under same administrative control.

SEE ALSO

`lpq`(1), `lpr`(1), `lprm`(1) in the *User's Reference*; `lpc`(1M); `syslog`(3C) in the *Programmer's Reference*.

NAME

`lpfilter` - administer filters used with the LP print service

SYNOPSIS

```
lpfilter -f filter-name -F path-name
lpfilter -f filter-name -
lpfilter -f filter-name -i
lpfilter -f filter-name -x
lpfilter -f filter-name -l
```

DESCRIPTION

The `lpfilter` command is used to add, change, delete, and list a filter used with the LP print service. These filters are used to convert the content type of a file to a content type acceptable to a printer. One of the following options must be used with the `lpfilter` command: `-F` *path-name* (or `-` for standard input) to add or change a filter; `-i` to reset an original filter to its factory setting; `-x` to delete a filter; or `-l` to list a filter description.

The argument `all` can be used instead of a *filter-name* with any of these options. When `all` is specified with the `-F` or `-` option, the requested change is made to all filters. Using `all` with the `-i` option has the effect of restoring to their original settings all filters for which predefined settings were initially available. Using the `all` argument with the `-x` option results in all filters being deleted, and using it with the `-l` option produces a list of all filters.

Adding or Changing a Filter

The filter named in the `-f` option is added to the filter table. If the filter already exists, its description is changed to reflect the new information in the input.

The filter description is taken from the *path-name* if the `-F` option is given, or from the standard input if the `-` option is given. One of the two must be given to define or change a filter. If the filter named is one originally delivered with the LP print service, the `-i` option will restore the original filter description.

When an existing filter is changed with the `-F` or `-` option, items that are not specified in the new information are left as they were. When a new filter is added with this command, unspecified items are given default values. (See below.)

Filters are used to convert the content of a request into a data stream acceptable to a printer. For a given print request, the LP print service will know the following: the type of content in the request, the name of the printer, the type of the printer, the types of content acceptable to the printer, and the modes of printing asked for by the originator of the request. It will use this information to find a filter or a pipeline of filters that will convert the content into a type acceptable to the printer.

Below is a list of items that provide input to this command, and a description of each item. All lists are comma or space separated.

```
Input types: content-type-list
Output types: content-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: filter-type
Command: shell-command
Options: template-list
```

`Input types`

This gives the types of content that can be accepted by the filter. (The

default is `any`.)

`Output types`
: This gives the types of content that the filter can produce from any of the input content types. (The default is `any`.)

`Printer types`
: This gives the type of printers for which the filter can be used. The LP print service will restrict the use of the filter to these types of printers. (The default is `any`.)

`Printers`
: This gives the names of the printers for which the filter can be used. The LP print service will restrict the use of the filter to just the printers named. (The default is `any`.)

`Filter type`
: This marks the filter as a `slow` filter or a `fast` filter. Slow filters are generally those that take a long time to convert their input. They are run unconnected to a printer, to keep the printers from being tied up while the filter is running. If a listed printer is on a remote system, the filter type for it must have the value `slow`. Fast filters are generally those that convert their input quickly, or those that must be connected to the printer when run. These will be given to the interface program to run connected to the physical printer.

`Command`
: This specifies the program to run to invoke the filter. The full program pathname as well as fixed options must be included in the *shell-command*; additional options are constructed, based on the characteristics of each print request and on the `Options` field. A command must be given for each filter.

  The command must accept a data stream as standard input and produce the converted data stream on its standard output. This allows filter pipelines to be constructed to convert data not handled by a single filter.

`Options`
: This is a comma separated list of templates used by the LP print service to construct options to the filter from the characteristics of each print request listed in the table later.

  In general, each template is of the following form:

  keyword pattern = replacement

  The *keyword* names the characteristic that the template attempts to map into a filter specific option; each valid *keyword* is listed in the table below. A *pattern* is one of the following: a literal pattern of one of the forms listed in the table, a single asterisk (`*`), or a regular expression. If *pattern* matches the value of the characteristic, the template fits and is used to generate a filter specific option. The *replacement* is what will be used as the option.

Regular expressions are the same as those found in the `ed`(1) or `vi`(1) commands. This includes the `\(...\)` and `\`*n* constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the `&`, which can be used to copy the entire *pattern* into the *replacement*.

The *replacement* can also contain a `*`; it too, is replaced with the entire *pattern*, just like the `&` of `ed`(1).

lp Option	Characteristic	*keyword*	Possible *patterns*
-T	Content type (input)	INPUT	*content-type*
N/A	Content type (output)	OUTPUT	*content-type*
N/A	Printer type	TERM	*printer-type*
-d	Printer name	PRINTER	*printer-name*
-f, -o cpi=	Character pitch	CPI	*integer*
-f, -o lpi=	Line pitch	LPI	*integer*
-f, -o length=	Page length	LENGTH	*integer*
-f, -o width=	Page width	WIDTH	*integer*
-P	Pages to print	PAGES	*page-list*
-S	Character set	CHARSET	*character-set-name*
	Print wheel	CHARSET	*print-wheel-name*
-f	Form name	FORM	*form-name*
-y	Modes	MODES	*mode*
-n	Number of copies	COPIES	*integer*

For example, the template

```
MODES landscape = -l
```

shows that if a print request is submitted with the `-y landscape` option, the filter will be given the option `-l`. As another example, the template

```
TERM * = -T *
```

shows that the filter will be given the option `-T` *printer-type* for whichever *printer-type* is associated with a print request using the filter.

As a last example, consider the template

```
MODES prwidth\=\(.*\) = -w\1
```

Suppose a user gives the command

```
lp -y prwidth=10
```

From the table above, the LP print service determines that the `-y` option is handled by a `MODES` template. The `MODES` template here works because the *pattern* `prwidth\=\(.*\)` matches the `prwidth=10` given by the user. The *replacement* `-w\1` causes the LP print service to generate the filter option `-w10`.

If necessary, the LP print service will construct a filter pipeline by concatenating several filters to handle the user's file and all the print options. (See `sh`(1) for a description of a pipeline.) If the print service constructs a filter pipeline, the `INPUT` and `OUTPUT` values used for each filter in the pipeline are the types of the input and output for that filter, not for the entire pipeline.

Deleting a Filter

The `-x` option is used to delete the filter specified in *filter-name* from the LP filter table.

Listing a Filter Description

The `-l` option is used to list the description of the filter named in *filter-name*. If the command is successful, the following message is sent to standard output:

```
Input types: content-type-list
Output types: content-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: filter-type
Command: shell-command
Options: template-list
```

If the command fails, an error message is sent to standard error.

SEE ALSO

lpadmin(1M).
lp(1) in the *User's Reference Manual*.

NAME

lpforms - administer forms used with the LP print service

SYNOPSIS

lpforms -f *form-name options*
lpforms -f *form-name* -A *alert-type* [-Q *minutes*] [-W *requests*]

DESCRIPTION

The lpforms command is used to administer the use of preprinted forms, such as company letterhead paper, with the LP print service. A form is specified by its *form-name*. Users may specify a form when submitting a print request [see lp(1)]. The argument all can be used instead of *form-name* with either of the command lines shown above. The first command line allows the administrator to add, change, and delete forms, to list the attributes of an existing form, and to allow and deny users access to particular forms. The second command line is used to establish the method by which the administrator is alerted that the form *form-name* must be mounted on a printer.

With the first lpforms command line, one of the following options must be used:

-F *pathname*	To add or change form *form-name*, as specified by the information in *pathname*
-	To add or change form *form-name*, as specified by the information from standard input
-x	To delete form *form-name* (this option must be used separately; it may not be used with any other option)
-l	To list the attributes of form *form-name*

Adding or Changing a Form

The -F *pathname* option is used to add a new form, *form-name*, to the LP print service, or to change the attributes of an existing form. The form description is taken from *pathname* if the -F option is given, or from the standard input if the - option is used. One of these two options must be used to define or change a form. *Pathname* is the path name of a file that contains all or any subset of the following information about the form.

Page length: *scaled-decimal-number*$_1$
Page width: *scaled-decimal-number*$_2$
Number of pages: *integer*
Line pitch: *scaled-decimal-number*$_3$
Character pitch: *scaled-decimal-number*$_4$
Character set choice: *character-set/print-wheel* [mandatory]
Ribbon color: *ribbon-color*
Comment:
comment
Alignment pattern: [*content-type*]
content

The term "scaled-decimal-number" refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a "trailing" letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing c); numbers that show sizes in inches (marked with a trailing i); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch,

or characters per inch.

Except for the last two lines, the above lines may appear in any order. The `Comment:` and *comment* items must appear in consecutive order but may appear before the other items, and the `Alignment pattern:` and the *content* items must appear in consecutive order at the end of the file. Also, the *comment* item may not contain a line that begins with any of the key phrases above, unless the key phrase is preceded with a > sign. Any leading > sign found in the *comment* will be removed when the comment is displayed. Case distinctions in the key phrases are ignored.

When this command is issued, the form specified by *form-name* is added to the list of forms. If the form already exists, its description is changed to reflect the new information. Once added, a form is available for use in a print request, except where access to the form has been restricted, as described under the `-u` option. A form may also be allowed to be used on certain printers only.

A description of each form attribute is below:

`Page length and Page Width`
: Before printing the content of a print request needing this form, the generic interface program provided with the LP print service will initialize the physical printer to handle pages *scaled-decimal-number*$_1$ long, and *scaled-decimal-number*$_2$ wide using the printer type as a key into the `terminfo` database.

The page length and page width will also be passed, if possible, to each filter used in a request needing this form.

`Number of pages`
: Each time the alignment pattern is printed, the LP print service will attempt to truncate the *content* to a single form by, if possible, passing to each filter the page subset of 1-*integer*.

`Line pitch` and `Character pitch`
: Before printing the content of a print request needing this form, the interface programs provided with the LP print service will initialize the physical printer to handle these pitches, using the printer type as a key into the `terminfo` database. Also, the pitches will be passed, if possible, to each filter used in a request needing this form. *Scaled-decimal-number*$_3$ is in lines per centimeter if a `c` is appended, and lines per inch otherwise; similarly, *scaled-decimal-number*$_4$ is in characters per centimeter if a `c` is appended, and characters per inch otherwise. The character pitch can also be given as `elite` (12 characters per inch), `pica` (10 characters per inch), or `compressed` (as many characters per inch as possible).

`Character set choice`
: When the LP print service alerts an administrator to mount this form, it will also mention that the print wheel *print-wheel* should be used on those printers that take print wheels. If printing with this form is to be done on a printer that has selectable or loadable character sets instead of print wheels, the interface programs provided with the LP print service will automatically select or load the correct character set. If `mandatory` is appended, a user is not allowed to select a different character set for use with the form; otherwise, the character set or print wheel named is a suggestion and a default only.

`Ribbon color`
: When the LP print service alerts an administrator to mount this form, it will

also mention that the color of the ribbon should be *ribbon-color*.

`Comment`
The LP print service will display the *comment* unaltered when a user asks about this form [see `lpstat`(1)].

`Alignment pattern`
When mounting this form an administrator can ask for the *content* to be printed repeatedly, as an aid in correctly positioning the preprinted form. The optional *content-type* defines the type of printer for which *content* had been generated. If *content-type* is not given, `simple` is assumed. Note that the *content* is stored as given, and will be readable only by the user `lp`.

When an existing form is changed with this command, items missing in the new information are left as they were. When a new form is added with this command, missing items will get the following defaults:

Page Length: `66`
Page Width: `80`
Number of Pages: `1`
Line Pitch: `6`
Character Pitch: `10`
Character Set Choice: `any`
Ribbon Color: `any`

Deleting a Form

The `-x` option is used to delete the form *form-name* from the LP print service.

Listing Form Attributes

The `-l` option is used to list the attributes of the existing form *form-name*. The attributes listed are those described under **Adding and Changing a Form,** above. Because of the potentially sensitive nature of the alignment pattern, only the administrator can examine the form with this command. Other people may use the `lpstat` command to examine the non-sensitive part of the form description.

Allowing and Denying Access to a Form

The `-u` option, followed by the argument `allow:`*login-ID-list* or `-u deny:`*login-ID-list* lets you determine which users will be allowed to specify a particular form with a print request. This option can be used with the `-F` or `-` option, each of which is described above under `Adding or Changing a Form.`

The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name`!`*login-ID*	A user on system *system_name*
system_name`!all`	All users on system *system_name*
`all!`*login-ID*	A user on all systems
`all`	All users on all systems

The LP print service keeps two lists of users for each form: an "allow-list" of people allowed to use the form, and a "deny-list" of people that may not use the form. With the `-u allow` option, the users listed are added to the allow-list and removed from the deny-list. With the `-u deny` option, the users listed are added to the deny-list and removed from the allow-list. (Both forms of the `-u` option can be run together with the `-F` or the `-` option.)

If the allow-list is not empty, only the users in the list are allowed access to the form, regardless of the contents of the deny-list. If the allow-list is empty but the deny-list is not, the users in the deny-list may not use the form, (but all others may use it). All users can be denied access to a form by specifying `-f deny:all`. All users can be allowed access to a form by specifying `-f allow:all`. (This is the default.)

Setting an Alert to Mount a Form

The `-f` *form-name* option is used with the `-A` *alert-type* option to define an alert to mount the form when there are queued jobs which need it. If this option is not used to arrange alerting for a form, no alert will be sent for that form.

The method by which the alert is sent depends on the value of the *alert-type* argument specified with the `-A` option. The *alert-types* are:

`mail` Send the alert message via the `mailx` command to the administrator.

`write` Write the message, via the `write` command, to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is arbitrarily chosen.

`quiet` Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the form *form-name* has been mounted and subsequently unmounted, messages will again be sent when the number of print requests reaches the threshold specified by the `-Q` option.

`none` Do not send messages until the `-A` option is given again with a different *alert-type* (other than `quiet`).

shell-command
Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. Note that the `mail` and `write` values for this option are equivalent to the values `mail` *login-ID* and `write` *login-ID* respectively, where *login-ID* is the current name for the administrator. This will be the login name of the person submitting this command unless he or she has used the `su` command to change to another login-ID. If the `su` command has been used to change the user ID, then the *user-name* for the new ID is used.

`list` Display the type of the alert for the form on standard output. No change is made to the alert.

The message sent appears as follows:

```
The form form-name needs to be mounted
on the printer(s):
printer (integer1 requests).
integer2 print requests await this form.
Use the ribbon-color ribbon.
Use the print-wheel print wheel, if appropriate.
```

The printers listed are those that the administrator had earlier specified were candidates for this form. The number $integer_1$ listed next to each printer is the number of requests eligible for the printer. The number $integer_2$ shown after the list of printers is the total number of requests awaiting the form. It will be less than the sum of the other numbers if some requests can be handled by more than one printer. The *ribbon-color* and *print-wheel* are those specified in the form description. The last line

in the message is always sent, even if none of the printers listed use print wheels, because the administrator may choose to mount the form on a printer that does use a print wheel.

Where any color ribbon or any print wheel can be used, the statements above will read:

```
Use any ribbon.
Use any print-wheel.
```

If *form-name* is `any`, the alerting defined in this command applies to any form for which an alert has not yet been defined. If *form-name* is `all`, the alerting defined in this command applies to all forms.

If the `-W` option is not given, the default procedure is that only one message will be sent per need to mount the form. Not specifying the `-W` option is equivalent to specifying `-W once` or `-W 0`. If *minutes* is a number greater than 0, an alert will be sent at intervals specified by *minutes*.

If the `-Q` option is also given, the alert will be sent when a certain number (specified by the argument *requests*) of print requests that need the form are waiting. If the `-Q` option is not given, or the value of *requests* is `1` or `any` (which are both the default), a message is sent as soon as anyone submits a print request for the form when it is not mounted.

Listing the Current Alert

The `-f` option, followed by the `-A` option and the argument `list` is used to list the type of alert that has been defined for the specified form *form-name*. No change is made to the alert. If *form-name* is recognized by the LP print service, one of the following lines is sent to the standard output, depending on the type of alert for the form.

- When *requests* `requests are queued:`
  `alert with` *shell-command* `every` *minutes* `minutes`
- When *requests* `requests are queued:`
  `write to` *user-name* `every` *minutes* `minutes`
- When *requests* `requests are queued:`
  `mail to` *user-name* `every` *minutes* `minutes`
- `No alert`

The phrase `every` *minutes* `minutes` is replaced with `once` if *minutes* (`-W` *minutes*) is 0.

Terminating an Active Alert

The `-A quiet` option is used to stop messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the form has been mounted and then unmounted, messages will again be sent when the number of print requests reaches the threshold *requests*.

Removing an Alert Definition

No messages will be sent after the `-A none` option is used until the `-A` option is given again with a different *alert-type*. This can be used to permanently stop further messages from being sent as any existing alert definition for the form will be removed.

SEE ALSO

`lpadmin`(1M), `terminfo`(4).
`lp`(1) in the *User's Reference Manual*.

NAME

lpprint, xlpprint - menu-driven lp interface

SYNOPSIS

lpprint

xlpprint [*X11-options*]

DESCRIPTION

The lpprint and xlpprint commands provide menu-driven interfaces to the lp command.

lpprint is designed for use on an ASCII terminal or terminal emulator. This version of the command presents you with menus and scrolling interactive queries to help you use the new functionality provided with the lp command.

xlpprint uses the X11 window system on a graphics workstation. When using this version of the command, you may use the mouse to select functions to be performed from menus, and dialog windows appear in which information is accepted.

Both interfaces are designed to be consistent while taking advantage of the capabilities of the display device. In both cases you follow the menus to the operation you wish to carry out. Once you make this selection, [x]lpprint prompts you to enter whatever information is necessary to carry out the operation. When all information has been obtained, the information is verified, you are asked to confirm that you want the operation carried out, and then the operation is performed.

Options

Options to xlpprint uses only those options which are defined by the X11 window system. See X(1).

EXAMPLES

```
# xlpprint -display mystation:0
```

This command runs the program with interactions taking place on the workstation mystation. (The -display option would not be needed if you were running the command on mystation.)

ENVIRONMENT

LANG Used to determine the *locale* (the default locale is C).

HOME Used to locate files that are expected in the user's home directory.

PATH Used to locate standard system commands, and certain utility programs.

SEE ALSO

idi(1), idl(4), X(1),
and the online help found when using xlpprint and lpprint.

NAME

lpsched, lpshut, lpmove – start/stop the LP print service and move requests

SYNOPSIS

/usr/lib/lp/lpsched
lpshut
lpmove *requests dest*
lpmove $dest_1$ $dest_2$

DESCRIPTION

Lpsched starts the LP print service; this can be done only by root or lp.

lpshut shuts down the print service. All printers that are printing at the time lpshut is invoked will stop printing. When lpsched is started again, requests that were printing at the time a printer was shut down will be reprinted from the beginning.

lpmove moves requests that were queued by lp between LP destinations. The first form of the lpmove command shown above (under **SYNOPSIS**) moves the named *requests* to the LP destination *dest*. *Requests* are request-IDs as returned by lp. The second form of the lpmove command will attempt to move all requests for destination $dest_1$ to destination $dest_2$; lp will then reject any new requests for $dest_1$.

Note that when moving requests, lpmove never checks the acceptance status [see accept(1M)] of the new destination. Also, the request-IDs of the moved request are not changed, so that users can still find their requests. The lpmove command will not move requests that have options (content type, form required, and so on) that cannot be handled by the new destination.

If a request was originally queued for a class or the special destination any, and the first form of lpmove was used, the destination of the request will be changed to *new-destination*. A request thus affected will be printable only on *new-destination* and not on other members of the class or other acceptable printers if the original destination was any.

FILES

/var/spool/lp/*

SEE ALSO

accept(1M), lpadmin(1M).
enable(1), lp(1), lpstat(1) in the *User's Reference Manual*.

NAME

lpsystem – register remote systems with the print service

SYNOPSIS

lpsystem [-t *type*] [-T *timeout*] [-R *retry*] [-y "*comment*"] *system-name* [*system-name* ...]
lpsystem -l [*system-name* ...]
lpsystem -r *system-name* [*system-name* ...]
lpsystem -A

DESCRIPTION

The lpsystem command is used to define parameters for the LP print service, with respect to communication (via a high-speed network such as STARLAN or TCP/IP) with remote systems. Only a privileged user (that is, the owner of the login root) may execute the lpsystem command.

Specifically, the lpsystem command is used to define remote systems with which the local LP print service can exchange print requests. These remote systems are described to the local LP print service in terms of several parameters that control communication: type, retry and timeout. These parameters are defined in /etc/lp/Systems. You can edit this file with a text editor (such as vi) but editing is not recommended.

The *type* parameter defines the remote system as one of two types: s5 (System V Release 4) or bsd (SunOS). The default type is s5.

The *timeout* parameter specifies the length of time (in minutes) that the print service should allow a network connection to be idle. If the connection to the remote system is idle (that is, there is no network traffic) for *N* minutes, then drop the connection. (When there is more work the connection will be restablished.) Legal values are n, 0, and *N*, where *N* is an integer greater than 0. The value n means "never time out"; 0 means "as soon as the connection is idle, drop it." The default is n.

The *retry* parameter specifies the length of time to wait before trying to re-establish a connection to the remote system, when the connection was dropped abnormally (that is, a network error). Legal values are n, 0, and *N*, where *N* is an integer greater than 0 and it means "wait *N* minutes before trying to reconnect. (The default is 10 minutes.) The value n means "do not retry dropped connections until there is more work"; 0 means "try to reconnect immediately."

The *comment* argument allows you to associate a free form comment with the system entry. This is visible when lpsystem -l is used.

System-name is the name of the remote system from which you want to be able to receive jobs, and to which you want to be able to send jobs. If the remote system has multiple names due to having multiple network interfaces, this should be the name that the hostname command, executed on the remote system returns.

The command lpsystem -l [*system-name*] will print out a description of the parameters associated with *system-name* (if a system has been specified), or with all the systems in its database (if *system-name* has not been specified).

The command lpsystem -r *system-name* will remove the entry associated with *system-name*. The print service will no longer accept jobs from that system or send jobs to it, even if the remote printer is still defined on the local system.

The command lpsystem -A will print out the TCP/IP address of the local machine in a format to be used when configuring the local port monitor to accept requests from a SunOS system.

NOTES

With respect to `/etc/lp/Systems`, this information is relatively minimal with respect to controlling network communications. For more information on network addresses and services, see *Managing TCP/IP on the DG/UX System*. Port monitors handle listening for remote service requests and routing the connection to the print service (see *Managing the DG/UX System*).

If the `Netconfig` and `Netdir` facilities are not set up properly, out-bound remote print service probably will not work. Similarly, if the local port monitors are not set up to route remote print requests to the print service, then service for remote systems will not be provided. (See "Allowing Remote Systems to Access Local Printers" and "Configuring a Local Port Monitor" in the "Print Service" chapter of the *System Administrator's Guide* to find out how to do this.)

With respect to the semantics of the *timeout* and *retry* values, the print service uses one process for each remote system with which it communicates, and it communicates with a remote system only when there is work to be done on that system or work being sent from that system.

The system initiating the connection is the "master" process and the system accepting the connection is the "slave" process. This designation serves only to determine which process dies (the slave) when a connection is dropped. This helps prevent there from being more than one process communicating with a remote system. Furthermore, all connections are bi-directional, regardless of the master/slave designation. You cannot control a system's master/slave designation. Now, keeping all this information in mind, if a master process times out, then both the slave and master will exit. If a slave times out, then it is possible that the master may still live and retry the connection after the retry interval. Therefore, one system's resource management strategy can effect another system's strategy.

With respect to `lpsystem -A`: a SunOS system (described with `-t bsd`) can be connected to your system only via TCP/IP, and print requests from a SunOS system can come in to your machine only via a special port (515). The address given to you from `lpsystem` will be the address of your system and port 515. This address is used by your TCP/IP port monitor (see `sacadm`(1M) and `nlsadmin`(1M)) to "listen" on that address and port, and to route connections to the print service. (This procedure is discussed in the "Service Access" chapter of the *System Administrator's Guide*.) The important point here is that this is where you get the address refered to in that procedure.

The command `lpsystem -A` will not work if your system name and IP address are not listed in `/etc/inet/hosts` and the printer service is not listed in `/etc/inet/services`.

FILES

`/var/spool/lp/* /etc/lp/*`

SEE ALSO

`netconfig`(4)
Managing TCP/IP on the DG/UX System
Managing the DG/UX System.

NAME

`lpusers` - set printing queue priorities

SYNOPSIS

```
lpusers -d priority-level
lpusers -q priority-level -u login-ID-list
lpusers -u login-ID-list
lpusers -q priority-level
lpusers -l
```

DESCRIPTION

The `lpusers` command is used to set limits to the queue priority level that can be assigned to jobs submitted by users of the LP print service.

The first form of the command (with `-d`) sets the system-wide priority default to *priority-level*, where *priority-level* is a value of 0 to 39, with 0 being the highest priority. If a user does not specify a priority level with a print request [see `lp`(1)], the default priority is used. Initially, the default priority level is 20.

The second form of the command (with `-q` and `-u`) sets the default highest *priority-level* (0-39) that the users in *login-ID-list* can request when submitting a print request. The *login-ID-list* argument may include any or all of the following constructs:

login-ID	A user on any system
system_name`!`*login-ID*	A user on the system *system_name*
system_name`!all`	All users on system *system_name*
`all!`*login-ID*	A user on all systems
`all`	All users on all systems

Users that have been given a limit cannot submit a print request with a higher priority level than the one assigned, nor can they change a request already submitted to have a higher priority. Any print requests submitted with priority levels higher than allowed will be given the highest priority allowed.

The third form of the command (with `-u`) removes any explicit priority level for the specified users. User names must be specific. `all` is not acceptable.

The fourth form of the command (with `-q`) sets the default highest priority level for all users not explicitly covered by the use of the second form of this command.

The last form of the command (with `-l`) lists the default priority level and the priority limits assigned to users.

SEE ALSO

`lp`(1) in the *User's Reference Manual*.

NAME

lsd - load a system dump from tape

SYNOPSIS

lsd [-s] [-d *dir_name*] *input_dev*

where:

dir_name The directory into which the dump files will be loaded; the default is the current directory.

input_dev The pathname of a no-rewind tape device that will be used to load the dump.

DESCRIPTION

Lsd loads the contents of a system dump tape set that was produced after a kernel panic or hang. Multi-tape sets are supported; lsd prompts for the next tape when it needs to change volumes.

A dump tape set normally has two tape files: the memory image and a cpio archive. If the memory image is too large for a single tape, it will be broken up into separate tape files across as many tapes as are necessary to store it.

Options are:

-s Specify that the cpio archive is not present on the tape or should be skipped.

-d Load the dump files into the specified directory. If *dir_name* does not exist, lsd creates it.

EXAMPLE

```
lsd /dev/rmt/0n
```

DIAGNOSTICS

Error diagnostics are routed to standard error; other diagnostics are routed to standard output. All are self-explanatory.

SEE ALSO

crash(1M), *Installing the DG/UX System*, and *Managing the DG/UX System*.

NOTE

The memory image (ordinarily named main_memory) consists of the contents of the system's physical memory at the time of the panic or hang. The cpio archive consists of files that were appended to the final dump tape after the system dump concluded. Usually, the archive contains the system's kernel image (ordinarily named dgux) and any test programs or files relevant to the system panic or hang. If both the memory image and the kernel image are available, the crash(1M) command can be used for post-mortem debugging of the system.

NAME

mail_pipe – invoke recipient command for incoming mail

SYNOPSIS

mail_pipe [-x *debug_level*] -r *recipient* -R *path_to_sender* -c *content_type* -S *subject*

DESCRIPTION

When a new mail message arrives, the mail command first checks if the recipient's mailbox indicates that the message is to be forwarded elsewhere (to some other recipient or as the input to some command). If the message is to be piped into a recipient-specified command, mail invokes mail_pipe to do some validation and then execute the command in the context of the recipient.

Command-line arguments are:

-x *debug_level*	Turn on debugging for this invocation. See the description of the -x option for the mail command for details.
-r *recipient*	The recipient's login id.
-R *path_to_sender*	The return address to the message's originator.
-c *content_type*	The value of the Content-Type: header line in the message.
-S *subject*	The value of the Subject: header line in the message if present.

mail_pipe is installed as a setuid-to-root process, thus enabling itself to change it's user and group ids to that of the recipient as necessary.

When invoked, mail_pipe performs the following steps (if a step fails, the exit code is noted as [*N*]):

- Validate invocation arguments [1].
- Verify that recipient name is $\leq$ 14 characters long [2].
- Verify that the setgid flag for the recipient mailbox is set [3].
- Open /var/mail/*recipient* [4].
- Verify that recipient's mailbox starts with the string Forward to [5].
- Find pipe symbol indicating start of command string in recipient mailbox [6].
- Find entry for recipient in /etc/passwd [7].
- Set gid to recipient's gid [8].
- Set uid to recipient's uid [9].
- Change current directory to recipient's login directory [10].
- Allocate space to hold newly exec'ed environment for recipient command [11].
- Parse the recipient command, performing any %*keyword* expansions required. See the 'Forwarding mail' section of mail(1), for more information regarding %*keyword* substitutions [12].
- Execute recipient command [13 if exec fails, otherwise exit code from recipient command itself].

FILES

/etc/passwd	to identify sender and locate recipients
/var/mail/*recipient*	incoming mail for *recipient*; that is, the mail file
/tmp/MLDBG*	debug trace file
/usr/lib/mail/mail_pipe	mail_pipe program

SEE ALSO

mail(1), notify(1), vacation(1)

NAME

mailstats – print sendmail statistics

SYNOPSIS

mailstats [-c] [-f *file*]

DESCRIPTION

Mailstats is used to collect statistics compiled by sendmail. Statistics include, for each mailer defined in the sendmail.cf file, number of messages to that mailer, number of kilobytes to that mailer, number of messages from that mailer, number of kilobytes from that mailer. The mailer is identified by its position in the sendmail.cf file - the first mailer defined is listed by mailstats as mailer 0, the second mailer defined as mailer 1, etc. Statistics are printed for all mailers with non-zero *msgsfr or msgsto* values.

Sample output:

```
Statistics from Sun Apr 28 10:25:17 1991

M msgsfr bytes_from  msgsto   bytes_to
0      1          1K      0          0K
2      0          0K      1          1K
```

In the example above, the stats accumulated since April 28 are shown. One message to sendmail has been received from mailer 0, and sendmail has sent one message to mailer 2.

Options are:

-c Clear/initialize the accumulated statistics, and reset the date value. This can only be done by a superuser (uid 0).

-f Use an alternate statistics file.

FILES

/etc/mailstats.st Contains statistics collected by sendmail and stored (in binary form) for mailstats.

/etc/sendmail.cf Contains the mailer definitions.

SEE ALSO

sendmail(1C).

WARNINGS

If the sendmail.cf file is changed to include new mailer definitions (or change the relative locations of the old mailer definitions) mailstats -c should be run to reset the counters.

BUGS

Granularity of byte counts transferred is very low. *Bytes_from* and *bytes_to* are rounded up to the nearest one kilobyte value - which means a one byte message and a 1000 byte messages are both counted as a kilobyte. Thus these statistics should be used as only a general indication of traffic through the various mailers.

NAME

makedbm - make a Network Information Service dbm file

SYNOPSIS

makedbm [-b] [-s] [-i *yp_input_file*] [-o *yp_output_name*]
[-d *yp_domain_name*] [-m *yp_master_name*] *infile outfile*

makedbm [-u *dbmfilename*]

DESCRIPTION

makedbm takes *infile* and converts it to a pair of files in ndbm(3C) format, namely *outfile*.pag and *outfile*.dir. Each line of the input file is converted to a single dbm record. All characters up to the first TAB or SPACE form the key, and the rest of the line is the data. If a line ends with '\', then the data for that record is continued on to the next line. It is left for the clients of the Network Information Service to interpret #; makedbm does not itself treat it as a comment character. *infile* can be '-', in which case the standard input is read.

makedbm is meant to be used in generating dbm files for the Network Information Service, and it generates a special entry with the key *yp_last_modified*, which is the date of *infile* (or the current time, if *infile* is '-').

OPTIONS

-b Interdomain. Propagate a map to all servers using the interdomain name server named(1M).

-s Secure map. Accept connections from secure NIS networks only.

-i Create a special entry with the key *yp_input_file*.

-o Create a special entry with the key *yp_output_name*.

-d Create a special entry with the key *yp_domain_name*.

-m Create a special entry with the key *yp_master_name*. If no master host name is specified, *yp_master_name* will be set to the local host name.

-u Undo a dbm file. That is, print out a dbm file one entry per line, with a single space separating keys from values.

EXAMPLE

It is easy to write shell scripts to convert standard files such as /etc/passwd to the key value form used by makedbm. For example,

```
#!/bin/awk -f
BEGIN { FS = ":"; OFS = "\t"; }
{ print $1, $0 }
```

takes the /etc/passwd file and converts it to a form that can be read by makedbm to make the Network Information Service file passwd.byname. That is, the key is a username, and the value is the entire line in the /etc/passwd file.

SEE ALSO

named(1M), yppasswd(1), ndbm(3C).

NAME

mkfifo - make FIFO special file

SYNOPSIS

mkfifo *path* ...

DESCRIPTION

Mkfifo creates the FIFO special files named by its argument list. The arguments are taken sequentially, in the order specified; and each FIFO special file is either created completely or, in the case of an error or signal, not created at all.

For each *path* argument, the mkfifo command behaves as if the function mkfifo [see mkfifo(3C)] was called with the argument *path* set to *path* and the *mode* set to the bitwise inclusive OR of S_IRUSR, S_IWUSR, S_IRGRP, S_IWGRP, S_IROTH and S_IWOTH.

If errors are encountered in creating one of the special files, mkfifo writes a diagnostic message to the standard error and continues with the remaining arguments, if any.

DIAGNOSTICS

mkfifo returns exit code 0 if all FIFO special files were created normally; otherwise it prints a diagnostic and returns a value greater than 0.

SEE ALSO

mkfifo(3C).

NAME

mkfs, newfs - create a file system

SYNOPSIS

/usr/sbin/mkfs [-m *free_space*] [-r *region_size*] [-i *inode_density*] [-s *data_element_log*] [-x *index_element_log*] [-S *dir_data_element_log*] [-X *dir_index_element_log*] [-e *first_anniversary_size*] [-E *second_anniversary_size*] [*density*] ["*pc*"] *special* [*proto*] [*gap blocks_per_cyl*]]

DESCRIPTION

Mkfs creates an empty file system on a logical or physical disk. The argument *special* must be a block-special or character-special device node, such as those nodes found in /dev/dsk or /dev/rdsk. The file system will span the entire disk; to create a file system of a particular size, first create a logical disk of that size with diskman(1M) and then run mkfs on that disk. Most invocations of mkfs will not need to alter the defaults, so no option arguments need to be specified:

```
# /usr/sbin/mkfs special
```

Newfs is identical to mkfs and is retained for Berkeley compatibility.

A floppy can be DOS formatted by using the following syntax:

```
# /usr/sbin/mkfs density "pc" special
```

where *density* specifies the capacity of the floppy to be formatted and *special* is a floppy-type device. "*pc*" indicates to *mkfs* that the floppy should be formatted so it can be used with the DFM file system manager (a file system that can read and write PC DOS floppies). There are two different floppy drives supported in DG/UX. The valid densities for the 5.25 inch drive are 360kb and 1220kb. The valid densities for the 3.50 inch drive are 720kb and 1440kb. *Mkfs* will do a hard format of the floppy and then lay down the file system format. If "*pc*" is not specified, it will create a DG/UX file system on the floppy. If "*pc*" is specified, then all the DG/UX file system options to mkfs will be ignored.

Other DG/UX file system arguments are:

-m *free_space:* The minimum percentage of free space the file system must have. If the file system's free space drops below this level, only a superuser can allocate more space. The value for free space must be an integer in the range 0 to 99, inclusive. The default value is 10%.

-r *region_size*: Determines how many blocks each Disk Allocation Region (DAR) in the file system will occupy (including the bitmap, inode table, and data blocks). This number must be an integer greater than or equal to 4032; the default value is based on the size of the file system. The last DAR created may be smaller than all others due to the target logical disk being an uneven multiple of DAR size. Each DAR (except the last one) is required to be large enough to hold the DAR bitmap, at least 64 inodes, and at least one default sized data element for files.

-i *inode_density*: Determines how many inode slots (potential files) the file system will have. The value specified is the ratio of usable data bytes in the logical disk to the number of inodes; the default is 3500. Any integer greater than zero may be specified, but the actual density will be rounded down to an integral multiple of 64 inodes per DAR. The maximum possible number of inodes occurs when every usable block of the DAR is occupied by inode

slots, except for the required space mention under *region_size*.

`-s` *data_element_log*: Determines the default data element size of files to be created in the new file system. The value specified is the element size in disk blocks, expressed as a base 2 logarithm. This number must be an integer from 0 to 31, inclusive. The default value is 4 (meaning data elements of 16 blocks).

`-x` *index_element_log*: Determines the default index element size of files to be created in the new file system. The value specified is the element size in disk blocks, expressed as a base 2 logarithm. This number must be an integer from 0 to 15, inclusive; the default value is 0 (meaning index elements of 1 block).

`-S` *dir_data_element_log*: Determines the default data element size of directories to be created in the new file system. The value specified is the element size in disk blocks, expressed as a base 2 logarithm. This number must be an integer from 0 to 31, inclusive; the default value is 4 (meaning data elements of 16 blocks).

`-X` *dir_index_element_log*: Determines the default index element size of directories to be created in the new file system. The value specified is the element size in disk blocks, expressed as a base 2 logarithm. This number must be an integer from 0 to 15, inclusive; the default value is 0 (meaning index elements of 1 block).

`-e` *first_anniversary_size*: Determines the maximum number of blocks a file can allocate in its initial disk allocation region before subsequent allocation requests are redirected to a different region. This number must be a positive integer; the default is determined based on the size of the disk allocation region.

`-E` *second_anniversary_size*: Determines the maximum number of blocks a file can allocate in any noninitial disk allocation region before subsequent allocation requests are redirected to a different region. This number must be a positive integer greater than first_anniversary_size; the default is determined based on the size of the disk allocation region.

special This is the name of the disk upon which a file system is to be created. `Special` must be the pathname of a writable character-special or block-special file.

proto If the argument following *special* is a name of a file that can be opened, it is taken as the pathname of a prototype file.

gap If this argument is specified, it is completely ignored. Under System V it is used to allow for characteristics of the target physical disk, a purpose that is irrelevant under the DG/UX system.

blocks_per_cyl
If this argument is specified, it is completely ignored. Under System V it is used to allow for characteristics of the target physical disk, a purpose that is irrelevant under the DG/UX system.

Prototype File Format

The prototype file format is as follows. The file contains tokens separated by spaces or new lines. The first token is the name of the bootstrap program; this is completely ignored since `mkfs` does not need to install bootstraps.

The second token is the size of the file system in disk blocks. DG/UX file systems must occupy the entire logical disk, so if this number is not equal to the disk size, `mkfs` will fail.

The third token is the number of inodes to be created in the file system. The specified number will be rounded up so that each DAR is given an equal number (which is itself a multiple of 64) of inodes.

The next set of tokens compose the specification for the root directory: the mode, the user id, the group id and the initial contents. The syntax of the contents field depends on the file mode. The mode token for a file is a six-character string. The first character specifies the file type using the same rules as `ls`(1). The second character is either "u" or "-" to specify setuid or not. The third character is either "g" or "-" to specify setgid or not. The rest of the mode is a 3 digit octal number in the same manner as `ls`(1). Two decimal number tokens follow the mode; they specify the user and group ids of the file's owner.

If the file is an ordinary file, the next token is a pathname from which the contents and size are copied. If the file is a block-special or character special file, two decimal tokens follow which give the file's major and minor device numbers. If the file is a directory, `mkfs` makes the entries specified. This specification may be recursive; each directory is terminated with the token "$".

DIAGNOSTICS

`Mkfs` will have no output except for diagnostic output in the case of errors. `mkfs` will return an exit status of 0 if and only if the specified file system was successfully created. Otherwise, `mkfs` will return 1.

SEE ALSO

`dfm`(4), `diskman`(1M), `fsck`(1M), `tunefs`(1M), `fs`(4).

NAME

mknod – build a special file

SYNOPSIS

/etc/mknod *name* b|c *major minor*

/etc/mknod *name* p

DESCRIPTION

Mknod makes a directory entry and corresponding inode for a device node or FIFO special file. Arguments are:

name Name of the entry.

b or c Indicator that the device node is block-type (e.g., disks) or character-type (other devices). Only the superuser may make device nodes.

major Number specifying the major device type in octal or decimal. Octal numbers must begin with the digit 0.

minor Number specifying the minor device (e.g., unit, drive, or line number) in decimal or octal.

p Indicator that mknod is to create FIFOs (named pipes). Any user may create FIFO nodes.

Major device numbers are assigned dynamically by the system as devices are configured. Minor device numbers are allocated by each device driver. Take note that, in general, it should never be necessary to use the mknod command, because all the devices configured into your kernel will automatically have nodes created for them each time your system is booted.

SEE ALSO

config(1M), mknod(2), master(4), system(4), *Customizing the DG/UX System*.

NAME

montbl – create monetary database

SYNOPSIS

montbl [-o *outfile*] *infile*
montbl -d [*file*]

DESCRIPTION

The montbl command takes as input a specification file, *infile*, that describes the formatting conventions for monetary quantities for a specific locale.

-o *outfile* Write the output on *outfile*; otherwise, write the output on a file named LC_MONETARY.

-d [*file*] Dump to standard output a text version of the LC_MONETARY table in file *file*. If no input file is specified, the monetary table in use for the current locale is dumped. You can modify the resulting text file, and use it as input to montbl, to produce a modified LC_MONETARY monetary table file. This file may be used to either replace the existing LC_MONETARY file in an existing locale, or to create a new locale. However, you must never modify any of the files (including LC_MONETARY) in /usr/lib/locale/C, the C locale.

The output of montbl (without -d) is suitable for use by the localeconv() function (see localeconv(3C)). Before *outfile* can be used by localeconv(), it must be installed in the /usr/lib/locale/*locale* directory with the name LC_MONETARY by someone who is super-user or a member of group bin. *locale* is the locale whose monetary formatting conventions are described in *infile*. This file must be readable by user, group, and other; no other permissions should be set. To use formatting conventions for monetary quantities described in this file, use setlocale(3C) to change the locale for category LC_MONETARY (or LC_ALL) to *locale* [see setlocale(3C)].

Once installed, this file will be used by the localeconv() function to initialize the monetary specific fields of a structure of type struct lconv. For a description of each field in this structure, see localeconv(3C).

```
struct      lconv {
        char *decimal_point;          /* "." */
        char *thousands_sep;          /* "" (zero length string) */
        char *grouping;               /* "" */
        char *int_curr_symbol;  /* "" */
        char *currency_symbol;  /* "" */
        char *mon_decimal_point;/* "" */
        char *mon_thousands_sep;/* "" */
        char *mon_grouping;           /* "" */
        char *positive_sign;          /* "" */
        char *negative_sign;          /* "" */
        char int_frac_digits;         /* CHAR_MAX */
        char frac_digits;       /* CHAR_MAX */
        char p_cs_precedes;           /* CHAR_MAX */
        char p_sep_by_space;          /* CHAR_MAX */
        char n_cs_precedes;           /* CHAR_MAX */
        char n_sep_by_space;          /* CHAR_MAX */
        char p_sign_posn;       /* CHAR_MAX */
        char n_sign_posn;       /* CHAR_MAX */
};
```

The specification file specifies the value of each `struct lconv` member, except for the first three members, *decimal_point*, *thousands_sep*, and *grouping* which are set by the `LC_NUMERIC` category of `setlocale`(3C). Each member's value is given on a line with the following format:

keyword <white space> *value*

where *keyword* is identical to the `struct lconv` field name and *value* is a quoted string for those fields that are a `char *` and an integer for those fields that are an `int`. For example,

```
int_curr_symbol          "ITL."
int_frac_digits          0
```

will set the international currency symbol and the number of fractional digits to be displayed in an internationally formatted monetary quantity to `ITL.` and `0`, respectively.

Blank lines and lines starting with a `#` are taken to be comments and are ignored. A character in a string may be in octal or hex representation. For example, `\141` or `\x61` could be used to represent the letter 'a'. If there is no specification line for a given structure member, then the default 'C' locale value for that member is used (see the values in comments in the `struct lconv` definition above).

Given below is an example of what the specification file for Italy would look like:

```
# Italy

int_curr_symbol          "ITL."
currency_symbol          "L."
mon_decimal_point ""
mon_thousands_sep "."
mon_grouping      "\3"
positive_sign            ""
negative_sign            "-"
int_frac_digits          0
frac_digits       0
p_cs_precedes            1
p_sep_by_space           0
n_cs_precedes            1
n_sep_by_space           0
p_sign_posn       1
n_sign_posn       1
```

FILES

`/usr/lib/locale/`*locale*`/LC_MONETARY`
LC_MONETARY database for *locale*

`/usr/lib/locale/C/montbl_C`
input file used to construct `LC_MONETARY` in the default locale.

SEE ALSO

`localeconv`(3C), `setlocale`(3C) in the *Programmer's Reference Manual.*

NAME

mount, umount - mount and dismount filesystems

SYNOPSIS

```
mount [ -p ]
mount -a [ fnv ] [ -t type ]
mount [ -fnrv ] [ -t type ] [ -o options ] filesystem directory
mount [ -vfn ] [ -o options ] filesystem | directory

umount [ -t type ] [ -h host ]
umount -a [ v ]
umount [ -v ] filesystem | directory ...
```

DESCRIPTION

Use mount to mount file systems, or to display currently mounted file systems. Use umount to unmount file systems.

The mount command has four formats:

mount [-p] — With no arguments, it displays currently mounted file systems.

mount -a [options] — With no arguments but with the a(ll) switch, it mounts some or all of the file systems listed in the file /etc/fstab.

mount [options] *filesystem directory* — With two arguments, it mounts the named *filesystem* on the named *directory*.

mount [options] *filesystem* | *directory* — With one argument, it mounts the named *filesystem* or *directory*, using a matching command line in the file /etc/fstab.

The umount command has three formats:

umount [-t *type* -h *host*] — With no arguments, it unmounts file systems of the specified type or from the specified host, that are listed in the file /etc/mnttab

umount -a [v] — With no argument but with the a(ll) switch, it unmounts the file systems listed in the file /etc/mnttab.

umount [-v] *filesystem* | *directory* — With one argument, it unmounts the file system that is mounted from *filesystem*, or mounted on *directory*.

Arguments

The *filesystem* argument names the file system to be mounted or unmounted. It may be local or remote. To specify a local file system, enter for *filesystem* a pathname that resolves to a local resource, such as: a disk partition (a logical file system created with diskman), a tape or cdrom device, or an area of memory. To specify a remote (nfs) file system, enter the *filesystem* argument as *host:pathname*, where *host* is the remote host's name and *pathname* is a directory on the remote host.

The *directory* argument is the mount point: the pathname of a directory on the local system. The *directory* must already exist. Usually, the mount point should be an empty directory: if not empty, its contents are hidden while the *filesystem* is mounted on it.

If *directory* is a symbolic link, the *filesystem* is mounted on the resolution directory rather than on the symbolic link.

Options

`-p` Display the mounted filesystems in a format suitable for use in `/etc/fstab`.

`-a` All. Attempt to mount all the filesystems described in `/etc/fstab`. If a *type* argument is specified with `-t`, mount all filesystems of that type. Filesystems are not necessarily mounted in the order shown in `/etc/fstab`.

`-f` Fake an `/etc/mnttab` entry, but do not actually mount any filesystems.

`-n` Mount the filesystem without making an entry in `/etc/mnttab`.

`-v` Verbose. Display a message indicating each filesystem being mounted.

`-t` *type* Specify a filesystem type. The accepted types are `dg/ux`, `cdrom`, `dos`, `swap`, and `nfs`. See *options* below for the arguments relevant for each type; see `fstab`(4) for a more detailed description of these types.

`-r` Mount the specified filesystem read-only, even if the entry in `/etc/fstab` specifies that it is to be mounted read-write.

Physically write-protected, magnetic tape, and cdrom filesystems should be mounted read-only. If they are mounted read-write, errors occur when the system attempts to update access times, even if no write operation is attempted.

`-o` *options*

Specify filesystem option arguments—one or more comma-separated words from the list below. Some options are valid for all filesystem types, while others apply to a specific type only.

These option arguments are valid for all filesystem types:

`ro | rw`
: Allow read-only or read-write access. *Note*: `cdrom` file systems are mounted read-only regardless of this argument.

`nosuid`
: Setuid execution disallowed.

`noauto`
: If this filesystem is currently mounted read-only, do not mount it. If the filesystem is not currently mounted, display an error message.

The default for `cdrom` filesystems is ' `ro, suid` '. For all other types, the default is ' `rw, suid` '.

These option arguments are valid for `dg/ux` filesystems:

`fsync_on_close`
: Whenever a file in this mounted file system is closed, write its dirty pages to disk. This option decreases the likelihood of data loss in the event of a system crash, but may degrade performance.

`fsck_log_size=`*n*
: Log changes to system data in a manner that allows fast recovery by `fsck`. This option may degrade performance.

`ramdisk` Mount a memory-resident file system. See `mfs`(4) for a detailed description of memory-resident file systems.

If the `ramdisk` argument is present, the following three arguments are also allowed:

`use_wired_memory`
By default, the data in the memory file system is subject to being swapped to disk. Use this argument to prevent swapping.

`max_file_space=`*n*
The default number of blocks in a memory file system is 2048. Use this argument to set the maximum size to *n* blocks. No memory is allocated until it is actually used. If the `use_wired_memory` argument is present, *n* may exceed the available memory. If this happens, the system allocates as many blocks as there are available, up to *n*; it does not report an error.

`max_file_count=`*n*
The default number of file nodes that can be allocated in a memory file system is 16384. Use this argument to set the maximum to *n*. If the `use_wired_memory` argument is present, *n* may cause available memory to be exceeded. If this happens, the system allocates as many file nodes as available memory allows, up to *n* nodes; it does not report an error.

These option arguments are valid for `nfs` (`NFS`) filesystems:

`bg | fg` If the first attempt fails, retry in the background, or, in the foreground.

`secure` Requires clients to use a more secure protocol when accessing the directory. Secure RPC using DES Authentication is an additional feature that must be purchased separately from the DG/UX™ ONC™/NFS® product. You must have this feature to use the `secure` option.

`retry=`*n* The number of times to retry the mount operation.

`rsize=`*n* Set the read buffer size to *n* bytes.

`wsize=`*n* Set the write buffer size to *n* bytes.

`timeo=`*n* Set the NFS timeout to *n* tenths of a second.

`retrans=`*n* The number of NFS retransmissions.

`port=`*n* The server IP port number.

`soft | hard` Return an error if the server does not respond, or continue the retry request until the server responds.

`intr` Allow keyboard interrupts to kill (or signal) a process that is hung waiting for a response from a remote server.

`acregmin=`*n* Hold cached attributes for at least *n* seconds after file modification.

`acregmax=`*n* Hold cached attributes for no more than *n* seconds after file modification.

`acdirmin=`*n* Hold cached attributes for at least *n* seconds after directory update.

`acdirmax=`*n* Hold cached attributes for no more than *n* seconds after directory update.

`actimeo=`*n* Set *min* and *max* times for regular files and directories to *n* seconds.

Regular defaults are:

```
fg,retry=10000,timeo=7,retrans=3,port=NFS_PORT,hard,\
acregmin=3,acregmax=60,acdirmin=30,acdirmax=60
```

Defaults for `rsize` and `wsize` are set internally by the system kernel.

umount Options

`-h` *host* Unmount all filesystems listed in `/etc/mnttab` that are remote-mounted from *host*.

`-t` *type* Unmount all filesystems listed in `/etc/mnttab` that are of a given *type*.

`-a` Unmount all filesystems currently mounted (as listed in `/etc/mnttab`).

`-v` Verbose. Display a message indicating each filesystem being unmounted.

NFS FILESYSTEMS

Background vs. Foreground

Filesystems mounted with the `bg` option indicate that `mount` is to retry in the background if the server's mount daemon (`mountd`(1M)) does not respond. `mount` retries the request up to the count specified in the `retry=`*n* option. Once the filesystem is mounted, each NFS request made in the kernel waits `timeo=`*n* tenths of a second for a response. If no response arrives, the time-out is multiplied by `2` and the request is retransmitted. When the number of retransmissions has reached the number specified in the `retrans=`*n* option, a filesystem mounted with the `soft` option returns an error on the request; one mounted with the `hard` option prints a warning message and continues to retry the request.

Read-Write vs. Read-Only

Filesystems that are mounted `rw` (read-write) should use the `hard` option to prevent possible loss of data; and the `intr` option to enable keyboard interrupts.

File Attributes

The attribute cache retains file attributes on the client. Attributes for a file are assigned a time to be flushed. If the file is modified before the flush time, then the flush time is extended by the time since the last modification (under the assumption that files that changed recently are likely to change soon). There is a minimum and maximum flush time extension for regular files and for directories. Setting `actimeo=`*n* extends flush time by *n* seconds for both regular files and directories.

EXAMPLES

To mount a local disk:

```
mount /dev/dsk/usr /usr
```

To mount all DG/UX filesystems:

```
mount -at dg/ux
```

To mount a remote filesystem:

```
mount serv:/usr/src /usr/src
```

To mount a remote filesystem that is listed in `/etc/fstab`:

```
mount /usr/src
```

To hard mount a remote filesystem:

```
mount -o hard serv:/usr/src /usr/src
```

To save current mount state:
```
mount -p > /etc/fstab
```

To mount a memory file system (the name `/dev/mem_tmp` is arbitrary and will be created by the `mount` command):

```
mount -o ramdisk /dev/mem_tmp /mnt
```

To mount a memory file system using wired memory:
```
mount -o ramdisk,use_wired_memory /dev/mem_tmp2 /memory1
```

FILES

`/etc/mnttab`	table of mounted filesystems
`/etc/fstab`	table of filesystems mounted at boot

SEE ALSO

`fsck`(1M), `mountd`(1M), `nfsd`(1M), `dg_mount`(2), `mkdir`(2), `open`(2), `umount`(2), `fstab`(4), `fs`(4), `mfs`(4), `mnttab`(4), `/usr/include/sys/dg_mount.h`, `/usr/include/sys/nfs.h`.

NAME

mountd – NFS mount request server

SYNOPSIS

```
/usr/etc/rpc.mountd [ -n ]
```

DESCRIPTION

mountd is an RPC server that answers file system mount requests. It reads the file /etc/xtab, described in exports(4), to determine which file systems are available for mounting by which machines. It also provides information as to what file systems are mounted by which clients. This information can be printed using the showmount(1M) command.

OPTIONS

-n Do not check that the clients are root users. Though this option makes things slightly less secure, it does allow older versions (pre-3.0) of client NFS to work.

FILES

/etc/xtab

SEE ALSO

showmount(1M), exports(4).

NAME

mvdir - move a directory

SYNOPSIS

/usr/sbin/mvdir *dirname name*

DESCRIPTION

Mvdir moves directories within a file system. *Dirname* must be a directory. If *name* does not exist, it will be created as a directory. If *name* does exist, and is a directory, *dirname* will be created as *name/dirname*. *dirname* and *name* may not be on the same path; that is, one may not subordinate to the other. For example:

```
mvdir x/y x/z
```

is legal, but

```
mvdir x/y x/y/z
```

is not.

SEE ALSO

mkdir(1), mv(1).

NOTE

Only the super-user can use mvdir.

NAME

named – Internet domain name server

SYNOPSIS

named [-d *debug*] [-p *portnumber*] [[-b] *bootfile*]

DESCRIPTION

The named daemon is the name server program of the domain name system. When invoked without arguments, named reads the default boot file /etc/named.boot, reads any initial data, and listens for queries.

Options are as follows:

-d Print debugging information. A number after the d determines the amount and detail of debugging information printed.

-p Use a different port number. The default is port number 53.

-b Use an alternate *bootfile*.

Any additional argument is interpreted as the name of the boot file. The named boot file contains information about where the name server should get its initial data. The following is a small example:

```
;   boot file for name server
;
; type          domain          source file or host
;
domain          abc.com
primary         abc.com         named.boot
secondary       cc.abc.com      128.223.1.78 10.32.1.10
cache           .               root.cache
```

The first uncommented line specifies that abc.com is the domain for which the server is authoritative. The second line states that the file named.boot contains authoritative data for the domain abc.com. The file named.boot contains data in the master file format described in RFC 883 except that all domain names are relative to the origin; in this case, abc.com (see below for a more detailed description).

The next line specifies that all authoritative data under cc.abc.com is to be transferred from the name server at 128.223.1.78. If the transfer fails it will try 10.32.1.10 and continue trying the addresses listed on this line. You can list as many as ten addresses on the line. The secondary copy is also authoritative for the specified domain.

The fourth line specifies data in root.cache is to be placed in the cache (in this case, well known data such as locations of root domain servers). The file root.cache is in the same format as named.boot.

A boot file consists of entries of the form:

$INCLUDE *filename*
$ORIGIN *domain*
domain opt_ttl opt_class type resource_record_data

where *domain* is "." for root, "@" for the current origin, or a standard domain name. If *domain* is a standard domain name that does not end with ".", the current origin is appended to the domain. Domain names ending with "." are unmodified. The *opt_ttl* field is an optional integer number for the time-to-live field. It defaults to zero.

The *opt_class* field is the object address type; currently only one type is supported, `IN`, for objects connected to the DARPA Internet. The *type* field is one of the following tokens; the data expected in the *resource_record_data* field is in parentheses.

`A`	a host address (dotted quad)
`NS`	an authoritative name server (domain)
`MX`	a mail exchanger (domain)
`CNAME`	the canonical name for an alias (domain)
`SOA`	marks the start of a zone of authority (5 numbers (see RFC 883))
`MB`	a mailbox domain name (domain)
`MG`	a mail group member (domain)
`MR`	a mail rename domain name (domain)
`NULL`	a null resource record (no format or data)
`WKS`	a well know service description (not yet implemented)
`PTR`	a domain name pointer (domain)
`HINFO`	host information (cpu_type OS_type)
`MINFO`	mailbox or mail list information (request_domain error_domain)

NOTES

The following signals have the specified effect when sent to the server process using the `kill`(1) command.

SIGHUP	Causes server to read named.boot and reload database.
SIGINT	Dumps current data base and cache to `/var/adm/named_dump.db`
SIGUSR1	Turns on debugging; each SIGUSR1 increments debug level.
SIGUSR2	Turns off debugging completely.

FILES

`/etc/named.boot`	name server configuration boot file
`/etc/named.pid`	the process id
`/var/adm/named.log`	debug output
`/var/adm/named_dump.db`	dump of the name servers database

SEE ALSO

`kill`(1), `nslookup`(1M), `gethostbyname`(3N), `resolver`(3C), `signal`(3C), `resolv.conf`(4M).

NAME

ncheck – generate names from i-numbers

SYNOPSIS

/etc/ncheck [-i *numbers*] [-a] [-s] [*devpathname*]

DESCRIPTION

The ncheck command generates a pathname versus i-number list of all files in one or more filesystems. Without arguments, ncheck looks in the file /etc/checklist for a list of device pathnames whose filesystems it should check. As an alternative, you can specify a device pathname on the command line. A device pathname indicates a special device file in /dev. For example, /dev/dsk/usr indicates that you want to check the /usr file system.

In the report that ncheck produces, names of directory files are followed by /.

Options are:

-i Reduce the report to only those files whose i-numbers follow.

-a Allow printing of the names . and .., which are ordinarily suppressed.

-s Reduce the report to special files and files with set-user-ID mode. This is intended to discover concealed violations of security policy.

DIAGNOSTICS

When the file system structure is improper, ?? denotes the parent of a parentless file, and a pathname beginning with ... denotes a loop.

SEE ALSO

fsck(1M), sort(1).

NAME

netinit – build a network protocol stack

SYNOPSIS

```
netinit   input_directives_file
```

or

```
netinit
```

where:

input_directives_file contains a sequence of directives.

DESCRIPTION

Use the netinit command to build the TCP/IP protocol stack.

Building the protocol stack involves opening new Streams to the drivers that TCP/IP uses, pushing appropriate protocol modules, and linking together the appropriate drivers. When you use netinit, you start a non-active controller. You must build the TCP/IP protocol stack before you use an interface to transmit and receive packets. Run netinit as a server (daemon) that builds and configures an arbitrary Streams stack. The server is driven by an *input_directives_file* composed of many individual netinit directives. The file may be delivered to netinit in a file through the command line or it may be read from standard input. All netinit output is directed to standard error. The netinit command uses standard Streams linkages throughout.

netinit Directives

A netinit directive is a sequence of ASCII words delimited by spaces, the first of which is the keyword. You separate directives with newlines. Directives emulate function calls. Each directive is interpreted and executed as soon as it is read from the standard input. The result of executing a directive is returned as an ASCII status string through standard error. The string "OK" is returned when no errors occur during execution of the directive. Appropriate negative acknowledgements are returned under error conditions.

The Directive Vocabulary

The directive primitives are closely involved with Streams operations such as the I_LINK ioctl. This section focuses on the nature of the *input_directives_file*. It is beyond the scope of this manual page to explain Streams functionality.

The netinit command recognizes the following keywords: AS, OPEN, CLOSE, PUSH, POP, LINK, UNLINK, ATTACH, RENAME and RUN. When using these keywords, case is not important.

Use the OPEN keyword as follows

```
OPEN device
```

or

```
OPEN device AS name
```

This opens the Streams driver with pathname *device*. The netinit program retains the file descriptor for use in processing subsequent directives that refer to the given device. If the optional AS clause is supplied, subsequent netinit directives may refer to the opened device using the supplied *name* rather than the device pathname.

Use the CLOSE keyword as follows:

```
CLOSE device
```

If the program has an open Stream to the named *device*, it is closed.

Use the PUSH keyword as follows:

PUSH *device module*

This pushes the specified Streams *module* onto the open Stream to the named *device*. An error occurs if the *module* or the *device* does not exist, or if there is not an open Stream to the device.

Use the POP keyword as follows:

POP *device*

This pops the *module* associated with the named *device* that is nearest the Stream head from the Stream. An error occurs if there is no open Stream to the named *device*, or if no modules are in the Stream.

Use the LINK keyword as follows:

LINK *mux_device lower_device*

The open Stream to *lower_device* is linked beneath the *mux_device*. An error occurs if there are not open Streams to both the *mux_device* and the lower_device, or if the *mux_device* is not a Streams multiplexing driver.

Use the UNLINK keyword as follows:

UNLINK *mux_device lower_device*

Unlink the *lower_device* from under the *mux_device*. An error occurs if *lower_device* does not specify a device linked under a Streams multiplexing driver specified by *mux_device*.

Use the RUN keyword as follows:

RUN *program_name* [*arglist*] [< *input_device*] [> *output_device*] [&]

You must specify a pathname to an executable file as the *program_name*. The optional *arglist* is passed to the *program_name*; the input and output device specifications, if specified, must refer to open Streams.

Use the RENAME keyword as follows:

RENAME *mux_device lower_device* AS *label*

This verifies that the *lower_device* has been linked under a *mux_device*, and that the multiplexor device has not been linked. It then causes the *mux_device* to assign the string *label* to the lower stream identified by *lower_device*.

Use the ATTACH keyword as follows

ATTACH *mux_device lower_device*

This is a special directive to the multiplexing device *mux_device* to associate a device linked under the multiplexer with the upper stream that made the request.

EXAMPLES

The following example shows a typical *input_directives_file* that builds a TCP/IP stack containing a loopback and an inen network device.

```
open /dev/ip as ip
open /dev/loop0 as loop0
link ip loop0
rename ip loop0 as loop0
run ifconfig loop0 localhost
open /dev/inen0 as inen0
push inen0 arp
link ip inen0
rename ip inen0 as inen0
run ifconfig inen0 mav33 broadcast 128.222.8.255 netmask 0xFFFFFF00
```

SEE ALSO

ifconfig(1M), inen(7), hken(7), *STREAMS Programmer's Guide for the DG/UX System*.

NAME

netstat - Show status for DG/UX network parameters

SYNOPSIS

netstat [-Aainsrt] [*interval*]

DESCRIPTION

The netstat command symbolically displays the contents of various network-related data structures. The options are as follows:

-A Obsolete and ignored. This is equivalent to the default display.

-a The state of all sockets; normally, sockets used by server processes are not shown.

-i The state of interfaces that have been auto-configured (interfaces statically configured into a system but not located at boot time are not shown).

-n Network addresses as numbers (normally, netstat interprets addresses and tries to display them symbolically).

-s Per-protocol statistics.

-r The routing tables.

-t Shows the local and remote addresses, send and receive queue sizes (in bytes), protocol, and (optionally) the internal state of the protocol for active sockets. This is the default display.

When invoked with an *interval* argument, netstat continuously displays a running count of statistics related to network interfaces. This display shows two columns: one for all interfaces, and one for the first interface on the interface list. The first line of each screen of information contains a summary of activity since the system was last rebooted. The netstat command pauses the number of seconds indicated by *interval* before refreshing the screen. Subsequent lines of output show values accumulated over the preceding interval.

If a socket's address specifies a network but no specific host address, address formats are displayed in the form *host-port* or *network-port*. When the host and network addresses are specified, they are displayed symbolically according to the databases /etc/hosts and /etc/networks, respectively. If a symbolic name for an address is unknown, or if the -n option is specified, the address is printed in the Internet dot format. Unspecified or wildcard addresses and ports appear as *-.

The interface display provides a table of cumulative statistics on packets transferred, errors, and collisions. The network address (currently Internet-specific) of the interface and the maximum transmission unit (mtu) are also displayed.

The routing table display indicates the available routes and their status. Each route consists of a destination host or network and a gateway to use in forwarding packets. The flags field shows the state of the route (U if up), whether the route is to a gateway (G), or whether the route is to a particular host (H). (Routes with an H flag appear as the result of an ICMP redirect or someone using the route(1M) command with the host parameter.) Direct routes are created for each interface attached to the local host. The refcnt field gives the current number of active uses of the route. Connection-oriented protocols normally hold on to a single route during a connection; protocols without connections obtain a route, then discard it. The use field provides a count of the number of packets sent using that route. The interface entry indicates the network interface used for the route.

SEE ALSO

route(1M), hosts(4), networks(4), protocols(4), services(4).

NAME

newkey – create a new key in the publickey database

SYNOPSIS

newkey [-h *hostname*] [-u *username*]

DESCRIPTION

NOTE: Secure RPC using DES Authentication is an additional feature that must be purchased separately from the DG/UX™ ONC™/NFS® package. You must have this feature to use the command described in this manual page.

newkey is normally run by the network administrator on the NIS master machine in order to establish public keys for users and super-users on the network. These keys are needed for using secure RPC or secure NFS.

newkey will prompt for the login password of the given username and then create a new public/secret key pair in /etc/publickey encrypted with the login password of the given user.

Use of this program is not required: users may create their own keys using chkey(1).

OPTIONS

-u *username* Create a new public key for the given username. Prompts for the Network Information Service (NIS) password of the given username.

-h *hostname* Create a new public key for the super-user at the given hostname. Prompts for the root password of the given hostname.

SEE ALSO

chkey(1), keylogin(1), keyserv(1M) publickey(4).

NAME

nfsd – Network File System server

SYNOPSIS

/usr/etc/nfsd [*nservers*]

DESCRIPTION

nfsd starts the server programs (daemons) that handle client filesystem requests. *nservers* is the number of file system request servers to start. This number should be based on the load expected on this server. Eight is a good number for most server traffic, twelve for increasing server traffic.

When a file that is opened by a client is unlinked (by the server), a file with a name of the form .nfs*XXX* (where *XXX* is a number) is created by the client. When the open file is closed, the .nfs*XXX* file is removed. If the client crashes before the file can be closed, the .nfs*XXX* file is not removed.

FILES

.nfs*XXX* client machine pointer to an open-but-unlinked file

SEE ALSO

mountd(1M), exports(4).

NAME

nfsstat - display Network File System statistics

SYNOPSIS

nfsstat [-csnr]

DESCRIPTION

nfsstat displays statistical information about the NFS (Network File System) and RPC (Remote Procedure Call) interfaces to the kernel. If no options are given the default is

```
nfsstat -csnr
```

That is, display everything.

Options

-c Display client information. Only the client side NFS and RPC information will be printed. Can be combined with the -n and -r options to print client NFS or client RPC information only.

-s Display server information.

-n Display NFS information. NFS information for both the client and server side will be printed. Can be combined with the -c and -s options to print client or server NFS information only.

-r Display RPC information.

Displays

The server RPC display includes the fields:

calls	total number of RPC calls received
badcalls	total number of calls rejected
nullrecv	number of times no RPC packet was available when trying to receive
badlen	number of packets that were too short
xdrcall	number of packets that had a malformed header

The server NFS display shows the number of NFS calls received (calls) and rejected (badcalls), and the counts and percentages for the various calls that were made.

The client RPC display includes the following fields:

calls	total number of RPC calls sent
badcalls	total number of calls rejected by a server
retrans	number of times a call had to be retransmitted
badxid	number of times a reply did not match the call
timeout	number of times a call timed out
wait	number of times a call had to wait on a busy CLIENT handle
newcred	number of times authentication information had to be refreshed

The client NFS display shows the number of calls sent and rejected, as well as the number of times a CLIENT handle was received (nclget), the number of times a call had to sleep while awaiting a handle (nclsleep), as well as a count of the various calls and their respective percentages.

SEE ALSO

nfsd(1M), statd(1M), rpcinfo(1M).

NAME

nlsadmin - network listener service administration

SYNOPSIS

```
/usr/sbin/nlsadmin -x
/usr/sbin/nlsadmin [ options ] net_spec
/usr/sbin/nlsadmin [ options ] -N port_monitor_tag
/usr/sbin/nlsadmin -V
/usr/sbin/nlsadmin -c cmd | -o streamname [ -p modules ] \
   [ -A address | -D ] [ -R prognum:versnum ]
```

DESCRIPTION

nlsadmin is the administrative command for the network listener process(es) on a machine. Each network has at least one instance of the network listener process associated with it; each instance (and thus, each network) is configured separately. The listener process "listens" to the network for service requests, accepts requests when they arrive, and invokes servers in response to those service requests. The network listener process may be used with any network (more precisely, with any connection-oriented transport provider) that conforms to the transport provider specification.

nlsadmin can establish a listener process for a given network, configure the specific attributes of that listener, and start and kill the listener process for that network. nlsadmin can also report on the listener processes on a machine, either individually (per network) or collectively.

The list below shows how to use nlsadmin. In this list, *net_spec* represents a particular listener process. Specifically, *net_spec* is the relative path name of the entry under /dev for a given network (that is, a transport provider). *address* is a transport address on which to listen and is interpreted using a syntax that allows for a variety of address formats. By default, *address* is interpreted as the symbolic ASCII representation of the transport address. An *address* preceded by a \x will let you enter an address in hexadecimal notation. Note that *address* must appear as a single word to the shell and thus must be quoted if it contains any blanks.

Changes to the list of services provided by the listener or the addresses of those services are put into effect immediately.

nlsadmin may be used with the following combinations of options and arguments:

nlsadmin gives a brief usage message.

nlsadmin -x
: reports the status of all of the listener processes installed on this machine.

nlsadmin *net_spec*
: prints the status of the listener process for *net_spec*.

nlsadmin -q *net_spec*
: queries the status of the listener process for the specified network, and reflects the result of that query in its exit code. If a listener process is active, nlsadmin will exit with a status of 0; if no process is active, the exit code will be 1; the exit code will be greater than 1 in case of error.

nlsadmin -v *net_spec*
: prints a verbose report on the servers associated with *net_spec*, giving the service code, status, command, and comment for each. It also specifies the uid the server will run as and the list of modules to be

pushed, if any, before the server is started.

`nlsadmin -z` *service_code net_spec*
prints a report on the server associated with *net_spec* that has service code *service_code*, giving the same information as in the `-v` option.

`nlsadmin -q -z` *service_code net_spec*
queries the status of the service with service code *service_code* on network *net_spec*, and exits with a status of 0 if that service is enabled, 1 if that service is disabled, and greater than 1 in case of error.

`nlsadmin -l` *address net_spec*
changes or set the transport address on which the listener listens (the general listener service). This address can be used by remote processes to access the servers available through this listener (see the `-a` option, below).

If *address* is just a dash ("-"), `nlsadmin` will report the address currently configured, instead of changing it.

A change of address takes effect immediately.

`nlsadmin -t` *address net_spec*
changes or sets the address on which the listener listens for requests for terminal service but is otherwise similar to the `-l` option above. A terminal service address should not be defined unless the appropriate remote login software is available; if such software is available, it must be configured as service code 1 (see the `-a` option, below).

`nlsadmin -i` *net_spec*
initializes an instance of the listener for the network specified by *net_spec*; that is, creates and initializes the files required by the listener as well as starting that instance of the listener. Note that a particular instance of the listener should be initialized only once. The listener must be initialized before assigning addresses or services.

`nlsadmin -a` *service_code* [`-p` *modules*] [`-w` *name*] `-c` *cmd* `-y` *comment net_spec*
adds a new service to the list of services available through the indicated listener. *service_code* is the code for the service, *cmd* is the command to be invoked in response to that service code, comprised of the full path name of the server and its arguments, and *comment* is a brief (free-form) description of the service for use in various reports. Note that *cmd* must appear as a single word to the shell; if arguments are required the *cmd* and its arguments must be enclosed in quotation marks. The *comment* must also appear as a single word to the shell. When a service is added, it is initially enabled (see the `-e` and `-d` options, below).

Service codes are alphanumeric strings, and are administered by AT&T. The numeric service codes 0 through 100 are reserved for internal use by the listener. Service code 0 is assigned to the nlps server, which is the service invoked on the general listening address. In particular, code 1 is assigned to the remote login service, which is the service automatically invoked for connections to the terminal login address.

If the `-p` option is specified, then *modules* will be interpreted as a list of STREAMS modules for the listener to push before starting the service being added. The modules are pushed in the order they are specified. *modules* should be a comma-separated list of modules, with no white

space included.

If the `-w` option is specified, then *name* is interpreted as the user name from `/etc/passwd` that the listener should look up. From the user name, the listener obtains the user ID, the group ID(s), and the home directory for use by the server. If `-w` is not specified, the default is to use the user name `listen`.

A service must explicitly be added to the listener for each network on which that service is to be available. This operation will normally be performed only when the service is installed on a machine, or when populating the list of services for a new network.

`nlsadmin -r` *service_code net_spec*
removes the entry for the *service_code* from that listener's list of services. This is normally done only in conjunction with the deinstallation of a service from a machine.

`nlsadmin -e` *service_code net_spec*
`nlsadmin -d` *service_code net_spec*
enables or disables (respectively) the service indicated by *service_code* for the specified network. The service must previously have been added to the listener for that network (see the `-a` option, above). Disabling a service will cause subsequent service requests for that service to be denied, but the processes from any prior service requests that are still running will continue unaffected.

`nlsadmin -s` *net_spec*
`nlsadmin -k` *net_spec*
starts and kills (respectively) the listener process for the indicated network. These operations will normally be performed as part of the system startup and shutdown procedures. Before a listener can be started for a particular network, it must first have been initialized (see the `-i` option, above). When a listener is killed, processes that are still running as a result of prior service requests will continue unaffected.

Under the Service Access Facility, it is possible to have multiple instances of the listener on a single *net_spec*. In any of the above commands, the option `-N` *port_monitor_tag* may be used in place of the *net_spec* argument. This argument specifies the tag by which an instance of the listener is identified by the Service Access Facility. If the `-N` option is not specified (i.e., the *net_spec* is specified in the invocation), then it will be assumed that the last component of the *net_spec* represents the tag of the listener for which the operation is destined. In other words, it is assumed that there is at least one listener on a designated *net_spec*, and that its tag is identical to the last component of the *net_spec*. This listener may be thought of as the primary, or default, listener for a particular *net_spec*.

`nlsadmin` is also used in conjunction with the Service Access Facility commands. In that capacity, the following combinations of options can be used:

`nlsadmin -V`
writes the current version number of the listener's administrative file to the standard output. It is used as part of the `sacadm` command line when `sacadm` add a port monitor to the system.

`nlsadmin -c` *cmd* | `-o` *streamname* [`-p` *modules*] [`-A` *address* | `-D`] \
[`-R` *prognum* : *versnum*]

formats the port monitor-specific information to be used as an argument to pmadm(1M).

The -c option specifies the full path name of the server and its arguments. *cmd* must appear as a single word to the shell, and its arguments must therefor be surrounded by quotes.

The -o option specifies the full path name of a FIFO or named STREAM through which a standing server is actually receiving the connection.

If the -p option is specified, then *modules* will be interpreted as a list of STREAMS modules for the listener to push before starting the service being added. The modules are pushed in the order in which they are specified. *modules* must be a comma-separated list, with no white space included.

If the -A option is specified, then *address* will be interpreted as the server's private address. The listener will monitor this address on behalf of the service and will dispatch all calls arriving on this address directly to the designated service. This option may not be used in conjunction with the -D option.

If the -D option is specified, then the service is assigned a private address dynamically, that is, the listener will have the transport provider select the address each time the listener begins listening on behalf of this service. For RPC services, this option will be often be used in conjunction with the -R option to register the dynamically assigned address with the rpcbinder. This option may not be used in conjunction with the -A option.

When the -R option is specified, the service is an RPC service whose address, program number, and version number should be registered with the rpcbinder for this transport provider. This registration is performed each time the listener begins listening on behalf of ththe service. *prognum* and *versnum* are the program number and version number, respectively, of the RPC service.

nlsadmin may be invoked by any user to generate reports but all operations that affect a listener's status or configuration are restricted to privileged users.

The options specific to the Service Access Facility may not be mixed with any other options.

SEE ALSO

listen(1M), pmadm(1M), rpcbind(1M), sacadm(1M)
Network Programmer's Guide

NOTES

Dynamically assigned addresses are not displayed in reports as statically assigned addresses are.

NAME

nslookup – query name servers interactively

SYNOPSIS

nslookup [*host_to_find* | – [*server_to_use*]]

DESCRIPTION

Use the nslookup command to query domain name servers. The nslookup command has two modes: interactive and non-interactive. When you use nslookup in interactive mode, it allows you to query the name server for information about various hosts or domains or print a list of hosts in the domain. When you use nslookup in non-interactive mode, it prints the Internet address of a specified *host_to_find* (which can be a hostname or domain name). You can also specify which *server_to_use* to obtain the information.

Arguments

Enter interactive mode as follows:

a) Specify no command arguments (the default name server will be used), or

b) Specify a hyphen (–) as the first argument and the hostname of a name server as the second argument.

To enter non-interactive mode, specify the name of the host to be looked up as the first argument. Optionally, specify a name server as the second argument.

Interactive Commands

Once you enter interactive mode, nslookup presents the > prompt, at which you can enter any one of nslookup's interactive commands. You can interrupt interactive commands at any time by typing a control-C. To exit, type a control-D (EOF). The command line length must be less than 80 characters.

Note: An unrecognized interactive command will be interpreted as a host name.

host [*server*]
: Look up information for *host* using the current default server or using *server* if it is specified.

server *server*
lserver *server*
: Change the default server to *server*. You can fully qualify the *server* if you wish. Lserver uses the initial server to look up information about a domain while server uses the current default server. If an authoritative answer can't be found, the names of servers that might have the answer are returned.

root
: Changes the default server to the server for the root of the domain name space. Currently, the server nic.ddn.mil. is used. (This command is a synonym for the lserver nic.ddn.mil.) The name of the root server can be changed with the set root command.

ls *domain* [> *filename*]
ls *domain* [>> *filename*]
ls -a *domain* [> *filename*]
ls -a *domain* [>> *filename*]

`ls -h` *domain* [`>` *filename*]
`ls -h` *domain* [`>>` *filename*]

`ls -d` *domain* [`>` *filename*]
List the information available for *domain*. The default output contains host names and their Internet addresses. The `-a` option lists aliases of hosts in the domain. The `-h` option lists CPU and operating system information for the domain. The `-d` option lists all contents of a zone transfer. When output is redirected to a file, hash marks are printed for every 50 records received from the server. You must use a space to separate the redirect operator from the output file name.

`view` *filename*
Sorts and lists the output of previous `ls` command(s) with *more*(1).

`help`

`?` Prints a brief summary of commands.

`set` *keyword*[=*value*]
This command is used to change state information that affects the lookups. Valid keywords are:

`all` Prints the current values of the various options to `set`. Information about the current default server and host is also printed.

[no]debug
Turn debugging mode on. A lot more information is printed about the packet sent to the server and the resulting answer.
(Default = `nodebug`, abbreviation = [no]deb)

[no]d2
Turn exhaustive debugging mode on. Essentially all fields of every packet are printed.
(Default = `nod2`)

[no]defname
Append the default domain name to every lookup.
(Default = `defname`, abbreviation = [no]def)

[no]search
With defname, search for each name in parent domains of the current domain.
(Default = `search`)

domain=*name*
Change the default domain name to *name*. The default domain name is appended to all lookup requests if the `defname` option has been set. The search list is set to parents of the domain with at least two components in their names.
(Default = value in hostname or `/etc/resolv.conf`, abbreviation = do)

querytype=*value*

type=*value*
Change the type of information returned from a query to one of:

SOA	The start of authority for a domain
A	The host's Internet address (the default)
CNAME	The canonical name for an alias
HINFO	The host CPU and operating system type
MB	The mail box
MX	The mail exchanger
MG	The mail group member
MINFO	The mailbox or mail list information
MR	The mail rename domain name
NS	Nameserver for the named zone.

(Abbreviation = q)

[no]recurse
Tell the name server to query other servers if it does not have the information.
(Default = recurse, abbreviation = [no]rec)

retry=*number*
Set the number of retries to *number*. When a reply to a request is not received within a certain amount of time (changed with set timeout), the request is resent. The retry value controls how many times a request is resent before giving up.
(Default = 2, abbreviation = ret)

root=*host*
Change the name of the root server to *host*. This affects the root command.
(Default = nic.ddn.mil., abbreviation = ro)

timeout=*number*
Change the time-out interval for waiting for a reply to *number* seconds.
(Default = 10 seconds, abbreviation = t)

[no]vc Always use a virtual circuit when sending requests to the server.
(Default = novc, abbreviation = [no]v)

DIAGNOSTICS

If the lookup request was not successful, an error message is printed. Possible errors are:

Time-out
The server did not respond to a request after a certain amount of time (changed with set timeout=*value*) and a certain number of retries (changed with set retry=*value*).

No information
Depending on the query type set with the set querytype command, no information about the host was available, though the host name is valid.

Non-existent domain
The host or domain name does not exist.

Connection refused
The server is down or unreachable.

`Network is unreachable`
The connection to the name server could not be made at the current time.

`Server failure`
The name server found an internal inconsistency in its database and could not return a valid answer.

`Refused`
The name server refused to service the request.

The following error should not occur and it indicates a bug in the program.

`Format error`
The name server found that the request packet was not in the proper format.

FILES

`/etc/resolv.conf` initial domain name and name server addresses.

SEE ALSO

`named`(1M), `resolver`(3C), `resolv.conf`(4).

NAME

osysadm – menu-driven system administration program

SYNOPSIS

osysadm [*argument*]

DESCRIPTION

Only a superuser can use the osysadm command.

The osysadm program is a menu-driven set of procedures for doing system administration tasks. Menu screens with interactive queries help you choose and execute the commands to administer the system. When you select a function, represented by a menu selection, the osysadm program passes control to that function. The function queries you for information, confirms the information you supply, and carries out your request.

If you type osysadm without an argument, the top level menu of system administration subcommands is displayed. Select again to go to the function of your choice. If a subcommand is used as an argument to the osysadm command, the menu of the subcommand is displayed, that is, you go directly to that subcommand without seeing the Main Menu.

SUBCOMMANDS

Typing osysadm without an argument displays the Main Menu:

```
                    SYSADM MAIN MENU

 1 diskmgmt         Enter the Diskman program
 2 sysmgmt          System configuration management menu
 3 fsmgmt           File system management menu
 4 fileinfo         File information menu
 5 ttymgmt          TTY management menu
 6 lpmgmt           Line Printer management menu
 7 usermgmt         User management menu
 8 uucpmgmt         UUCP management menu
 9 networkmgmt      Network management menu
10 releasemgmt      Software release management menu
11 clientmgmt       Diskless and X terminal client management menu

Enter a number, a name, the initial part of a name,
? or number? for HELP, or q to QUIT:
```

Help?

Call up a help script on any menu item by typing the item number followed by a question mark. If you don't understand a query, type a question mark. For example, typing 7? will display a help message on usermgmt usage.

To Exit

To exit at any time, type q or Q. At the menu level, q takes you all the way out to the shell. At the query level, q takes you back to the previous menu. To exit temporarily to perform a shell command, type !*command*. For example, you could stop in the middle of interacting with lpmgmt dialogues and type !mailx diablo to send a mail message to diablo. At the conclusion of the mailx command, you are returned to your previous position in the osysadm program.

Navigating in Sysadm

Use the ^ entry to back up one menu at a time if you are at the menu selection level.

You can always get out with `q`, reinvoke the Main Menu by typing `osysadm`, then continue.

FILES

The files that support `osysadm` are found in `/usr/admin`.

SEE ALSO

`diskman`(1M), *Installing the DG/UX System*, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NOTE

The `osysadm` command has been replaced by a new `sysadm`. Support for `osysadm` will be removed in a future release.

NAME

passmgmt - password files management

SYNOPSIS

passmgmt -a *options name*
passmgmt -m *options name*
passmgmt -d *name*

DESCRIPTION

The passmgmt command updates information in the password files. This command works with the /etc/passwd file.

passmgmt -a adds an entry for user *name* to the password files. This command does not create any directory for the new user and the new login remains locked (with the string *new* in the password field) until the passwd(1) command is executed to set the password.

passmgmt -m modifies the entry for user *name* in the password files. All the fields (except the password field) in the /etc/passwd entry can be modified by this command. Only fields entered on the command line will be modified.

passmgmt -d deletes the entry for user *name* from the password files. It will not remove any files that the user owns on the system; they must be removed manually.

The following options are available:

-y Perform the requested operation on the global NIS (YP) database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-c *comment* A short description of the login. It is limited to a maximum of 128 characters and defaults to an empty field.

-h *homedir* Home directory of *name*. It is limited to a maximum of 256 characters and defaults to /home/*name*.

-u *uid* UID of the *name*. This number must range from 0 to the maximum non-negative value for the system. It defaults to the next available UID greater than 99. Without the -o option, it enforces the uniqueness of a UID.

-o This option allows a UID to be non-unique. It is used only with the -u option.

-g *gid* GID of the *name*. This number must range from 0 to the maximum non-negative value for the system. The default is 1.

-s *shell* Login shell for *name*. It should be the full pathname of the program that will be executed when the user logs in. The maximum size of *shell* is 256 characters. The default is for this field to be empty and to be interpreted as /usr/bin/sh.

-l *logname* This option changes the *name* to *logname*. It is used only with the -m option.

The total size of each login entry is limited to a maximum of 511 bytes in each of the password files.

FILES

/etc/passwd,
/etc/opasswd,
/etc/yp/Makefile

DIAGNOSTICS

The passmgmt command exits with one of the following values:

0 Success.

1 Permission denied.

2 Invalid command syntax. Usage message of the passmgmt command will be displayed.

3 Invalid argument provided to option.

4 UID in use.

6 Unexpected failure. Password files unchanged.

7 Unexpected failure. Password file(s) missing.

8 Password file(s) busy. Try again later.

9 *name* does not exist (if -m or -d is specified), already exists (if -a is specified), or logname already exists (if -m -l is specified).

SEE ALSO

useradd(1M), userdel(1M), usermod(1M), passwd(4).
passwd(1) in the *User's Reference Manual*.

NOTES

You cannot use a colon or carriage return as part of an argument because it is interpreted as a field separator in the password file.

This command will be removed in a future release. Its functionality has been replaced and enhanced by useradd, userdel, and usermod. These commands are currently available.

NAME

`ping` – Network debugging

SYNOPSIS

`/usr/bin/ping` *host* [*timeout*]

DESCRIPTION

The `ping` command tests whether a node on an Internet network is up and working, though the upper layers of TCP/IP need not be up. This program sends an ICMP echo packet to *host* using a RAW socket interface, expecting the required ICMP response. If the ICMP packet is sent and received correctly, then a message is printed saying that *host* is alive. If there are errors locating *host*, creating the socket, sending the message, or receiving the message, an error message is printed.

The `ping` continues testing the network until timeout seconds have elapsed or until an answer is received. The default timeout is 20 seconds. The *host* argument can be a name or an Internet address.

EXAMPLE

```
$ ping harpo
harpo is alive      (This line is returned.)
```

NAME

pkgadd - transfer software package to the system

SYNOPSIS

pkgadd [-d *device*] [-r *response*] [-n] [-a *admin*] [*pkginst1* [*pkginst2*[...]]]

pkgadd -s *spool* [-d *device*] [*pkginst1* [*pkginst2*[...]]]

DESCRIPTION

pkgadd transfers the contents of a software package from the distribution medium or directory to install it onto the system. Used without the -d option, pkgadd looks in the default spool directory for the package (var/spool/pkg). Used with the -s option, it reads the package to a spool directory instead of installing it.

-d Install or copy a package from *device*. *device* can be a full path name to a directory or the identifiers for tape, floppy disk or removable disk (for example, /var/tmp, /dev/pdsk/1, or diskette1). It can also be the device alias. getdev(1M) displays the list of valid device aliases.

-r Identify a file or directory, *response*, which contains output from a previous pkgask session. This file supplies the interaction responses that would be requested by the package in interactive mode. *response* must be a full pathname.

-n Install in non-interactive mode. The default mode is interactive.

-a Define an installation administration file, *admin*, to be used in place of the default administration file. The token none overrides the use of any *admin* file, and thus forces interaction with the user. Unless a full path name is given, pkgadd looks in the /var/sadm/install/admin directory for the file.

pkginst Specify the package instance or list of instances to be installed. The token all may be used to refer to all packages available on the source medium. The format *pkginst*.* can be used to indicate all instances of a package.

-s Read the package into the directory *spool* instead of installing it.

When executed without options, pkgadd users /var/spool/pkg (the default spool directory).

NOTES

When transferring a package to a spool directory, the -r, -n, and -a options cannot be used.

The -r option can be used to indicate a directory name as well as a filename. The directory can contain numerous *response* files, each sharing the name of the package with which it should be associated. This would be used, for example, when adding multiple interactive packages with one invocation of pkgadd. Each package would need a *response* file. If you create response files with the same name as the package (*i.e. package1* and *package2*), then name the directory in which these files reside after the -r.

The -n option will cause the installation to halt if any interaction is needed to complete it.

SEE ALSO

getdev(1M), installf(1M), pkgask(1M), pkgchk(1), pkgmk(1), pkginfo(1), pkgparam(1), pkgproto(1), pkgtrans(1), pkgrm(1M), putdev(1M), removef(1M).

NAME

pkgask - stores answers to a request script

SYNOPSIS

pkgask [-d *device*] -r *response pkginst* [*pkginst* [...]]

DESCRIPTION

pkgask allows the administrator to store answers to an interactive package (one with a request script). Invoking this command generates a *response* file that is then used as input at installation time. The use of this *response* file prevents any interaction from occurring during installation since the file already contains all of the information the package needs.

-d Runs the request script for a package on *device*. *device* can be a directory pathname or the identifiers for tape or removable disk (for example, /var/tmp and /dev/rmt/0). The default device is the installation spool directory.

-r Identifies a file or directory, which should be created to contain the responses to interaction with the package. The name must be a full pathname. The file, or directory of files, can later be used as input to the pkgadd command.

pkginst Specifies the package instance or list of instances for which request scripts will be created. The token all may be used to refer to all packages available on the source medium.

NOTES

The -r option can be used to indicate a directory name as well as a filename. The directory name is used to create numerous *response* files, each sharing the name of the package with which it should be associated. This would be used, for example, when you will be adding multiple interactive packages with one invocation of pkgadd. Each package would need a *response* file. To create multiple response files with the same name as the package instance, name the directory in which the files should be created and supply multiple instance names with the pkgask command. When installing the packages, you will be able to identify this directory to the pkgadd command.

SEE ALSO

installf(1M), pkgadd(1M), pkgchk(1), pkgmk(1), pkginfo(1), pkgparam(1), pkgproto(1), pkgtrans(1), pkgrm(1M), removef(1M).

NAME

pkgchk – check accuracy of installation

SYNOPSIS

pkgchk [-l|-acfqv] [-nx] [-p *path1*[,*path2* ...] [-i *file*] [*pkginst*...]

pkgchk -d *device* [-l|v] [-p *path1*[,*path2* ...] [-i *file*] [*pkginst*...]

pkgchk -m *pkgmap* [-e *envfile*] [-l|-acfqv] [-nx] [-i *file*]
[-p *path1*[,*path2* ...]]

DESCRIPTION

pkgchk checks the accuracy of installed files or, by use of the -l option, displays information about package files. The command checks the integrity of directory structures and the files. Discrepancies are reported on stderr along with a detailed explanation of the problem.

The first synopsis defined above is used to list or check the contents and/or attributes of objects that are currently installed on the system. Package names may be listed on the command line, or by default the entire contents of a machine will be checked.

The second synopsis is used to list or check the contents of a package which has been spooled on the specified device, but not installed. Note that attributes cannot be checked for spooled packages.

The third synopsis is used to list or check the contents and/or attributes of objects which are described in the indicated *pkgmap*.

The option definitions are:

-l Lists information on the selected files that make up a package. It is not compatible with the a, c, f, g, and v options.

-a Audits the file attributes only, does not check file contents. Default is to check both.

-c Audits the file contents only, does not check file attributes. Default is to check both.

-f Corrects file attributes if possible. If used with the -x option, it removes hidden files. When pkgchk is invoked with this option it creates directories, named pipes, links and special devices if they do not already exist.

-q Quiet mode. Does not give messages about missing files.

-v Verbose mode. Files are listed as processed.

-n Does not check volatile or editable files. This should be used for most post-installation checking.

-x Searches exclusive directories, looking for files which exist that are not in the installation software database or the indicated *pkgmap* file.

-p Only checks the accuracy of the pathname or pathnames listed. *pathname* can be one or more pathnames separated by commas (or by white space, if the list is quoted).

-i Reads a list of pathnames from *file* and compares this list against the installation software database or the indicated *pkgmap* file. Pathnames which are not contained in *inputfile* are not checked.

-d Specifies the device on which a spooled package resides. *device* can be a directory pathname or the identifiers for tape, floppy disk or removable disk (for example, /var/tmp or /dev/rmt/0).

`-m` Requests that the package be checked against the pkgmap file *pkgmap*.

`-e` Requests that the pkginfo file named as *envfile* be used to resolve parameters noted in the specified pkgmap file.

pkginst Specifies the package instance or instances to be checked. The format *pkginst*.* can be used to check all instances of a package. The default is to display all information about all installed packages.

SEE ALSO

`pkgadd`(1M), `pkgask`(1M), `pkginfo`(1), `pkgrm`(1M), `pkgtrans`(1).

NAME

pkgrm - removes a package from the system

SYNOPSIS

pkgrm [-n] [-a *admin*] [*pkginst1* [*pkginst2*[...]]]

pkgrm -s *spool* [*pkginst*]

DESCRIPTION

pkgrm will remove a previously installed or partially installed package from the system. A check is made to determine if any other packages depend on the one being removed. The action taken if a dependency exists is defined in the admin file.

The default state for the command is in interactive mode, meaning that prompt messages are given during processing to allow the administrator to confirm the actions being taken. Non-interactive mode can be requested with the -n option.

The -s option can be used to specify the directory from which spooled packages should be removed.

The options and arguments for this command are:

-n	Non-interactive mode. If there is a need for interaction, the command will exit. Use of this option requires that at least one package instance be named upon invocation of the command.
-a	Defines an installation administration file, *admin*, to be used in place of the default *admin* file.
-s	Removes the specified package(s) from the directory "spool."
pkginst	Specifies the package to be removed. The format *pkg_abbrev*.* can be used to remove all instances of a package.

SEE ALSO

installf(1M), pkgadd(1M), pkgask(1M), pkgchk(1), pkginfo(1), pkgmk(1), pkgparam(1), pkgproto(1), pkgtrans(1), removef(1M).

NAME

pmadm - port monitor administration

SYNOPSIS

```
pmadm -a [-p pmtag | -t type] -s svctag -i id -m pmspecific
      -v ver [-f xu] [-y comment] [-z script]

pmadm -r -p pmtag -s svctag

pmadm -e -p pmtag -s svctag

pmadm -d -p pmtag -s svctag

pmadm -l [-t type | -p pmtag] [-s svctag]

pmadm -L [-t type | -p pmtag] [-s svctag]

pmadm -g -p pmtag -s svctag [-z script]

pmadm -g -s svctag -t type -z script
```

DESCRIPTION

pmadm is the administrative command for the lower level of the Service Access Facility hierarchy, that is, for service administration. A port may have only one service associated with it although the same service may be available through more than one port. In order to uniquely identify an instance of a service the pmadm command must identify both the port monitor or port monitors through which the service is available (-p or -t) and the service (-s). See the option descriptions below.

pmadm performs the following functions:

- add or remove a service
- enable or disable a service
- install or replace a per-service configuration script
- print requested service information

Any user on the system may invoke pmadm to request service status (-l or -L) or to print per-service configuration scripts (-g without the -z option). pmadm with other options may be executed only by a privileged user.

The options have the following meanings:

-a Add a service. pmadm adds an entry for the new service to the port monitor's administrative file. Because of the complexity of the options and arguments that follow the -a option, it may be convenient to use a command script or the menu system to add services. If you use the menu system, enter sysadm ports, then choose the port_services option.

-d Disable a service. Add x to the flag field in the entry for the service *svctag* in the port monitor's administrative file. This is the entry used by port monitor *pmtag*. See the -f option, below, for a description of the flags available.

-e Enable a service. Remove x from the flag field in the entry for the service *svctag* in the port monitor administrative file. This is the entry used by port monitor *pmtag*. See the -f option, below, for a description of the flags available.

-f xu The -f option specifies one or both of the following two flags which are then included in the flag field of the entry for the new service in the port monitor's administrative file. If the -f option is not included, no flags are set and the default conditions prevail. By default, a new service is enabled and no utmp

entry is created for it. A `-f` option without a following argument is illegal.

x Do not enable the service *svctag* available through port monitor *pmtag*.

u Create a `utmp` entry for service *svctag* available through port monitor *pmtag*.

`-g` Print, install, or replace a per-service configuration script. The `-g` option with a `-p` option and a `-s` option prints the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-p` option, a `-s` option, and a `-z` option installs the per-service configuration script contained in the file *script* as the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-s` option, a `-t` option, and a `-z` option installs the file *script* as the per-service configuration script for service *svctag* available through any port monitor of type *type*. Other combinations of options with `-g` are invalid.

`-i id` `id` is the identity that is to be assigned to service *svctag* when it is started. `id` must be an entry in `/etc/passwd`.

`-l` The `-l` option requests service information. Used by itself and with the options described below it provides a filter for extracting information in several different groupings.

`-l` By itself, the `-l` option lists all services on the system.

`-l -p` *pmtag*
Lists all services available through port monitor *pmtag*.

`-l -s` *svctag*
Lists all services with tag *svctag*.

`-l -p` *pmtag* `-s` *svctag*
Lists service *svctag*.

`-l -t` *type* Lists all services available through port monitors of type *type*.

`-l -t` *type* `-s` *svctag*
Lists all services with tag *svctag* available through a port monitor of type *type*.

Other combinations of options with `-l` are invalid.

`-L` The `-L` option is identical to the `-l` option except that output is printed in a condensed format.

`-m` *pmspecific*
pmspecific is the port monitor-specific portion of the port monitor administrative file entry for the service.

`-p` *pmtag*
Specifies the tag associated with the port monitor through which a service (specified as `-s` *svctag*) is available.

`-r` Remove a service. When `pmadm` removes a service, the entry for the service is removed from the port monitor's administrative file.

`-s` *svctag*
Specifies the service tag associated with a given service. The service tag is assigned by the system administrator and is part of the entry for the service in the port monitor's administrative file.

`-t` *type*
Specifies the the port monitor type.

`-v` *ver* Specifies the version number of the port monitor administrative file. The version number may be given as

```
-v `pmspec -V`
```

where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm` for `ttymon` and `nlsadmin` for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a `-V` option.

`-y` *comment*
Associate *comment* with the service entry in the port monitor administrative file.

`-z` *script*
Used with the `-g` option to specify the name of the file that contains the per-service configuration script. Modifying a configuration script is a three-step procedure. First a copy of the existing script is made (`-g` alone). Then the copy is edited. Finally, the copy is put in place over the existing script (`-g` with `-z`).

OUTPUT

If successful, **pmadm** will exit with a status of `0`. If it fails for any reason, it will exit with a nonzero status.

Options that request information write the requested information to the standard output. A request for information using the `-l` option prints column headers and aligns the information under the appropriate headings. In this format, a missing field is indicated by a hyphen. A request for information in the condensed format using the `-L` option prints the information in colon-separated fields; missing fields are indicated by two successive colons. `#` is the comment character.

EXAMPLES

Add a service to a port monitor with tag `pmtag`. Give the service the tag `svctag`. Port monitor-specific information is generated by `specpm`. The service defined by `svctag` will be invoked with identity `root`.

```
pmadm -a -p pmtag -s svctag -i root -m `specpm -a arg1 -b arg2` \
   -v `specpm -V`
```

Add a service with service tag `svctag`, identity `guest`, and port monitor-specific information generated by `specpm` to all port monitors of type `type`:

```
pmadm -a -s svctag -i guest -t type -m `specpm -a arg1 -b arg2` \
     -v `specpm -V`
```

Remove the service `svctag` from port monitor `pmtag`:

```
pmadm -r -p pmtag -s svctag
```

Enable the service `svctag` available through port monitor `pmtag`:

```
pmadm -e -p pmtag -s svctag
```

Disable the service `svctag` available through port monitor `pmtag`:

```
pmadm -d -p pmtag -s svctag
```

List status information for all services:

```
pmadm -l
```

List status information for all services available through the port monitor with tag `ports`:

```
pmadm -l -p ports
```

List the same information in condensed format:

```
pmadm -L -p ports
```

List status information for all services available through port monitors of type `listen`:

```
pmadm -l -t listen
```

Print the per-service configuration script associated with the service `svctag` available through port monitor `pmtag`:

```
pmadm -g -p pmtag -s svctag
```

FILES

`/etc/saf/`*`pmtag`*`/_config`
`/etc/saf/`*`pmtag/svctag`*
`/var/saf/`*`pmtag`*`/*`

SEE ALSO

`sacadm`(1M), `sac`(1M), `doconfig`(3N).

NAME

pmtd - start the pseudo magnetic tape device server

SYNOPSIS

pmtd [-c *cachesize*]

DESCRIPTION

Only a superuser can start the pmtd server (daemon).

The pmtd server runs on the local machine and starts the execution of the pseudo magnetic tape device server, a server process which handles local requests to perform I/O operations on a tape device hosted by a remote machine.

A local user opens a special file in the directory /dev/pmt. The DG/UX kernel then communicates with the pmtd server to perform operations. Across the network, the pmtd server starts the execution of, and communicates with, another server process rmt(1), which performs the operations on the real tape device.

The pmtd server consults two files (which it reads every time a pmt(7) special file is opened) in providing its service. /etc/pmttapetab contains the necessary information to access the remote tape device. While performing I/O, if an error occurs at the remote end, pmtd will use /etc/pmterrtab in conjunction with /etc/pmttapetab to provide a semantically equivalent DG/UX errno value corresponding to the remote error. If pmtd cannot access /etc/pmterrtab or the remote error does not appear in the table, pmtd passes the remote error back unchanged unless the value is not a legal DG/UX errno value. In such cases, EIO will be passed back as a general catch-all.

The pmtd server has an optional argument, *cachesize,* which allows the user to specify the size (in bytes) of the server's internal cache. This cache is used as a buffer to hold data whenever the /etc/pmttapetab *cache* field is set to Y or N.

The pmtd server automatically puts itself in the background.

The protocol used between pmt(7) and pmtd is expressed below in the following BNF-like specification:

lifetime operation:

*<ctrl_pckt[data]reply_pckt[data]>**

The *ctrl_pckt* and *reply_pckt* packets have a fixed size of 32 bytes.

ctrl_pckt = <*cmd*<parm>**null*>
cmd = ele of { 'OP','CL','RD','WR','IO' }

reply_pckt = <*ok_reply*<null>> ||
<*err_reply*<null>>>
ok_reply = <'OK'*parm*>
err_reply = <'ER'*parm*>

parm = <<ele of (*sigma* - *nl*)>**nl*>
sigma = ele of { isprint()-able chars }
data = char*
nl = newline char
null = null char

ctrl_pckt's:

cmd *parm*'s (*nl*'s not shown) <meaning>

'OP' *device_name*<open_intent> open
'CL' *device_handle* close
'RD' *byte_cnt* read
'WR' *byte_cnt*<data> write
'IO' *cmd*<op_code>*op_cnt* ioctl

ok_reply's:

cmd returns *parm*'s (*nl*'s not shown)

'OP' none
'CL' none
'RD' *byte_cnt*<data>
'WR' *byte_cnt*
'IO' <io_code>

err_reply's:

cmd returns *parm*'s (*nl*'s not shown)

'OP' <err_num>
'CL' none
'RD' <err_num>
'WR' <err_num>
'IO' <err_num>

FILES

`/etc/pmttapetab`	Table with information about remote tape devices.
`/etc/pmterrtab`	Table of equivalent error numbers among different operating systems.
`/usr/include/sys/errno.h`	File describing DG/UX `errno` values.
`/usr/include/sys/mtio.h`	File describing DG/UX tape operations.

SEE ALSO

`rmt`(1), `close`(2), `ioctl`(2), `open`(2), `pmttapetab`(4), `pmterrtab`(4), `pmt`(7).

CAVEATS

All `ioctl`(2) calls with the command set to `MTIOCTOP` supported by the real tape device, are supported by the `pmtd` server if the `/etc/pmttapetab` entry has the *cache* field set to `N`.

For the following, assume that the `/etc/pmttapetab` entry has the *cache* field set to `Y`.

All BCS required operations are supported so long as the real tape device conforms to the BCS. An `ioctl`(2) call with the command set to `MTIOCTOP` is currently supported for the operations: `MTWEOF`, `MTFSF`, `MTBSF`, `MTREW`, `MTOFFL`, and `MTNOP`.

If a system call, other than `open`(2), fails on a `pmt`(7) special file, its state thereafter is not guaranteed until the special file is closed and reopened.

NAME

portmap - DARPA port to RPC program number mapper

SYNOPSIS

/usr/etc/rpc.portmap

DESCRIPTION

portmap is a server that converts RPC program numbers into DARPA protocol port numbers. It must be running in order to make RPC calls.

When an RPC server is started, it will tell portmap what port number it is listening to, and what RPC program numbers it is prepared to serve. When a client wishes to make an RPC call to a given program number, it will first contact portmap on the server machine to determine the port number where RPC packets should be sent.

Normally, standard RPC servers are started by inetd(1M), so portmap must be started before inetd is invoked.

SEE ALSO

inetd(1M), rpcinfo(1M), inetd.conf(4M).

BUGS

If portmap crashes, all servers must be restarted.

NAME

probedev – probe system for devices

SYNOPSIS

probedev [-f *device-table-file*]

where:

device-table-file Table of device names for which to probe.

DESCRIPTION

The probedev command probes the system for any of the devices listed in the *device-table-file*. Any device which is currently configured, or which can be successfully configured, is written to standard output.

Devices can be configured only if the device driver for the device is configured into the currently-running kernel.

Options are:

-f Read device names from *device-table-file*. The default is /usr/etc/probedevtab.

FILES

/usr/etc/probedevtab Default table of possible device names.

DIAGNOSTICS

The probedev command exits with one of the following exit codes:

0 The command completed successfully.

1 The command failed due to a missing *device-table-file*.

2 The command failed because the user is not super-user.

3 The command failed due to an error in the command line.

SEE ALSO

dg_sysctl(2).

NOTE

Devices which are installed at non-standard addresses will not be found by probedev.

You must be super-user to use this command.

NAME

prfld, prfstat, prfdc, prfsnap, prfpr - operating system profiler

SYNOPSIS

/usr/sbin/prfld [-f *device_name*] [*namelist*]
/usr/sbin/prfstat on
/usr/sbin/prfstat [-f *device_name*] off
/usr/sbin/prfdc [-f *device_name*] *file* [*period* [*off_hour*]]
/usr/sbin/prfsnap [-f *device_name*] *file*
/usr/sbin/prfpr *file* [*cutoff* [*namelist*]]

DESCRIPTION

Prfld, prfstat, prfdc, prfsnap, and prfpr form a system of programs to study activity on a DG/UX operating system.

Prfld is used to initialize the recording mechanism in the system. It generates a table containing the starting address of each system subroutine as extracted from *namelist*.

Prfstat is used to enable or disable the sampling mechanism. Prfstat also reveals the number of text addresses being measured.

Prfdc and prfsnap collect data by copying the current value of all the text address counters to a file for analysis. Prfdc stores the counters into *file* every *period* minutes and turns off at *off_hour* (valid values for *off_hour* are 0 - 24). Prfsnap collects data at the time of invocation only, appending the counter values to *file*.

Prfpr formats the data collected by prfdc or prfsnap. Each text address is converted to the nearest text symbol (as found in *namelist*) and is printed if the percentage activity for that range is greater than *cutoff*.

These commands accept the following option:

-f *device_name*
: This option allows profiling on a specific CPU rather than on all CPUs at once. Device_names indicate the processor as defined in the /dev directory. Valid device_names are of the form /dev/prf0, /dev/prf1...to /dev/prf*n*. To perform profiling on all processors specify /dev/prf, without a processor number.

FILES

/dev/prf*	Interface(s) to profile data and text addresses
/dgux	Default for namelist file

SEE ALSO

prf(7).

NAME

putdev - edit device table

SYNOPSIS

putdev -a *alias* [*attribute=value* [. . .]]

putdev -m *device* *attribute=value* [*attribute=value* [. . .]]

putdev -d *device* [*attribute* [. . .]]

where:

alias The alias of the device to be added.

device The pathname or alias of the device whose attribute is to be added, modified, or removed.

attribute A device attribute to be added or modified. It can be any of the device attributes described under NOTES except alias. This prevents an accidental modification or deletion of a device's alias from the table.

value The value to be assigned to a device's attribute.

DESCRIPTION

Putdev can add a new device to the device table, modify an existing device description or remove a device entry from the table. The first synopsis is used to add a device. The second synopsis is used to modify existing entries by adding or changing attributes. If a specified attribute is not defined, this option adds that attribute to the device definition. If it is already defined, it modifies the attribute definition. The third synopsis is used to delete either an entire device entry or, if the attribute argument is used, to delete an attribute assignment for a device.

Options

The options for this command are:

-a Adds a device to the device table using the specified attributes. The device must be referenced by its *alias*.

-m Modifies a device entry in the device table. If an entry already exists, it adds any specified attributes that are not defined. It also modifies any attributes which already have a value with the value specified with this command.

-d Removes a device from the device table, when executed without the *attributes* argument. Used with the *attribute* argument, it deletes the given attribute specification for *device* from the table.

Device Attributes

The following list shows all of the attributes which can be defined for a device:

alias The unique name by which a device is known. No two devices in the database may share the same alias name. The name is limited in length to 14 characters and should contain only alphanumeric characters and also the following special characters if they are escaped with a backslash: underscore (_), dollar sign ($), hyphen (-), and period (.).

bdevice The pathname to the block special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the cdevice field, if any. (It is the administrator's responsibility to ensure that these major/minor numbers are unique in the database.)

`capacity`	The capacity of the device or of the typical volume, if removable.
`cdevice`	The pathname to the character special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the `bdevice` field, if any. (It is the administrator's responsibility to ensure that these `major/minor` numbers are unique in the database.)
`cyl`	Used by the command specified in the `mkfscmd` attribute.
`desc`	A description of any instance of a volume associated with this device (such as floppy diskette).
`dpartlist`	The list of disk partitions associated with this device. Used only if `type=disk`. The list should contain device aliases, each of which must have `type=dpart`.
`dparttype`	The type of disk partition represented by this device. Used only if `type=dpart`. It should be either `fs` (for filesystem) or `dp` (for data partition).
`erasecmd`	The command string that, when executed, erases the device.
`fmtcmd`	The command string that, when executed, formats the device.
`fsname`	The filesystem name on the file system administered on this partition, as supplied to the `/usr/sbin/labelit` command. This attribute is specified only if `type=dpart` and `dparttype=fs`.
`gap`	Used by the command specified in the `mkfscmd` attribute.
`mkfscmd`	The command string that, when executed, places a file system on a previously formatted device.
`mountpt`	The default mount point to use for the device. Used only if the device is mountable. For disk partitions where `type=dpart` and `dparttype=fs`, this attribute should specify the location where the partition is normally mounted.
`nblocks`	The number of blocks in the filesystem administered on this partition. Used only if `type=dpart` and `dparttype=fs`.
`ninodes`	The number of inodes in the filesystem administered on this partition. Used only if `type=dpart` and `dparttype=fs`.
`norewind`	The name of the character special device node that allows access to the serial device without rewinding when the device is closed.
`pathname`	Defines the pathname to an i-node describing the device (used for non-block or character device pathnames, such as directories).
`type`	A token that represents inherent qualities of the device. Standard types include: 9-track, ctape, disk, directory, diskette, dpart, and qtape.
`volname`	The volume name on the filesystem administered on this partition, as supplied to the `/usr/sbin/labelit` command. Used only if `type=dpart` and `dparttype=fs`.
`volume`	A text string used to describe any instance of a volume associated with this device. This attribute should not be defined for devices which are not removable.

FILES

`/etc/device.tab`

DIAGNOSTICS

The command will exit with one of the following values:

`0` = successful completion of the task.

`1` = command syntax incorrect, invalid option used, or internal error occurred.

`2` = device table could not be opened for reading or new device table could not be created.

`3` = if executed with the `-a` option, indicates that an entry in the device table with the alias *alias* already exits. If executed with the `-m` or `-d` options, indicates that no entry exists for device *device*.

`4` = indicates that `-d` was requested and one or more of the specified attributes were not defined for the device.

SEE ALSO

`devattr`(1M), `getdev`(1M), `putdgrp`(1M).

NAME

putdgrp - edit device group table

SYNOPSIS

putdgrp [-d] *dgroup* [*device* [. . .]]

DESCRIPTION

putdgrp modifies the device group table. It performs two kinds of modification. It can modify the table by creating a new device group or removing a device group. It can also change group definitions by adding or removing a device from the group definition.

When the command is invoked with only a *dgroup* specification, the command adds the specified group name to the device group table if it does not already exist. If the -d option is also used with only the *dgroup* specification, the command deletes the group from the table.

When the command is invoked with both a *dgroup* and a *device* specification, it adds the given device name (or names) to the group definition. When invoked with both arguments and the -d option, the command deletes the device name (or names) from the group definition.

When the command is invoked with both a *dgroup* and a *device* specification and the device group does not exist, it creates the group and adds the specified devices to that new group.

The options and arguments for this command are:

-d Deletes the group or, if used with *device*, the device from a group definition.

dgroup Specifies a device group name.

device Specifies the pathname or alias of the device that is to added to or deleted from the device group.

DIAGNOSTICS

The command will exit with one of the following values:

0 = successful completion of the task.

1 = command syntax incorrect, invalid option used, or internal error occurred.

2 = device group table could not be opened for reading or a new device group table could not be created.

3 = if executed with the -d option, indicates that an entry in the device group table for the device group *dgroup* does not exist and so cannot be deleted. Otherwise, indicates that the device group *dgroup* already exists and cannot be added.

4 = if executed with the -d option, indicates that the device group *dgroup* does not have as members one or more of the specified devices. Otherwise, indicates that the device group *dgroup* already has one or more of the specified devices as members.

EXAMPLE

To add a new device group:

```
putdgrp floppies
```

To add a device to a device group:

```
putdgrp floppies diskette2
```

To delete a device group:

```
putdgrp -d floppies
```

To delete a device from a device group:

```
putdgrp -d floppies diskette2
```

FILES

`/etc/dgroup.tab`

SEE ALSO

`getdgrp`(1M), `listdgrp`(1M), `putdev`(1M).

NAME

pwck, grpck – check password or group file

SYNOPSIS

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

where:

file The pathname of a password or group file; default = /etc/passwd for pwck, /etc/group for grpck.

DESCRIPTION

Pwck scans the password file *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-to-use-as-shell exist. The criteria for determining a valid login name are described in *Managing the DG/UX™ System*.

Grpck verifies all entries in the group file. This verification includes a check of the number of fields, group name, group ID, whether any login names belong to more than NGROUPS_MAX groups and that all login names appear in the password file. The fields for the password and login names may be empty.

FILES

/etc/group
/etc/passwd

DIAGNOSTICS

For each line with some inconsistency, the line is displayed followed by an explanation of the problem.

SEE ALSO

group(4), passwd(4), and *Managing the DG/UX™ System*.

NAME

reboot - restart the operating system

SYNOPSIS

/sbin/reboot [-lnq] [*boot-path*]

DESCRIPTION

Reboot restarts the kernel. The kernel is loaded into memory by the PROM monitor, which transfers control to it.

Although reboot can be run by the privileged user at any time, shutdown(1M) is normally used first to warn all users logged in of the impending loss of service. See shutdown(1M) for details.

The reboot command performs a sync(1) operation on the disks, and then a multiuser reboot is initiated. See init(1M) for details.

Reboot normally logs the reboot to the system log server, syslogd(1M), and places a shutdown record in the login accounting file /etc/wtmp. These actions are inhibited if the -l, -n, or -q options are present.

The following options are available:

-l Do not log the system shutdown to syslogd.

-n Do not sync the disks before halting (see sync(1M)). This is a dangerous option because data in system buffers may be lost.

-q Quick. Reboots quickly and ungracefully, without first shutting down running processes.

boot-path

Use the specified *boot-path* when rebooting. If the *boot-path* is not specified, the current boot path is used. By default, this is the boot path used when the system was last booted. The current boot path can be changed with the dg_sysctl(1M) command. If the *boot-path* is the empty string or spaces, the boot path saved by the System Control Monitor (SCM) is used.

Power Fail and Crash Recovery

Normally, the system will reboot itself at power-up or after crashes.

FILES

/etc/wtmp login accounting file

SEE ALSO

crash(1M), dg_sysctl(1M), fsck(1M), halt(1M), init(1M), shutdown(1M), sync(1M), syslogd(1M), in the *System Manager's Reference for the DG/UX System*.

Using the AViiON System Control Monitor (SCM).

NAME

removef - remove a file from software database

SYNOPSIS

removef *pkginst path1* [*path2* ...]

removef -f *pkginst*

DESCRIPTION

Removef informs the system that the user, or software, intends to remove a pathname. Output from removef is the list of input pathnames that may be safely removed (no other packages have a dependency on them).

After all files have been processed, removef should be invoked with the -f option to indicate that the removal phase is complete.

EXAMPLE

The following shows the use of removef in an optional pre-install script:

```
echo "The following files are no longer part of this package
      and are being removed."
removef $PKGINST /dev/xt[0-9][0-9][0-9] |
while read pathname
do
      echo "$pathname"
      rm -f $pathname
done
removef -f $PKGINST || exit 2
```

SEE ALSO

installf(1M), pkgadd(1M), pkgask(1M), pkgchk(1), pkginfo(1), pkgmk(1), pkgproto(1), pkgtrans(1), pkgparam(3X).

NAME

`restore` – incrementally restore a file system

SYNOPSIS

`/usr/sbin/restore` *key* [*filename* ...]

where:

key A character string composed of one function keyletter and zero or more optional keyletters

filename The name of a data file or directory specifying the files that are to be restored

DESCRIPTION

`Restore` reads files and symbolic links dumped with the `dump`(1M) or `dump2`(1M) commands. Its actions are controlled by a key argument. Unless the `h` key is specified (see below), the appearance of a directory name refers to the files and (recursively) subdirectories of that directory.

The function keyletters are:

`r` Read the tape and load its contents into the current directory. This keyletter should be used only to restore a complete dump tape onto a clear file system or to restore an incremental dump tape after a full level zero restore. Following is a typical sequence to restore a complete dump:

```
/usr/sbin/mkfs /dev/dsk/mnt
/sbin/mount /dev/dsk/mnt /mnt
cd /mnt
restore r
```

You can invoke `restore` again to get an incremental dump in on top of this. Note that `restore` leaves a file `restoresymtable` in the current directory to pass information between incremental passes by `restore`. This file should be removed when the last incremental tape has been restored.

`R` Request a particular tape of a multi-volume set on which to restart a full restoration (see the `r` key above). This lets you interrupt `restore`, then restart it.

`x` Extract the named files from the tape. If the named file matches a directory whose contents had been written onto the tape and the `h` key is not specified, the directory is recursively extracted. The owner, modification time, and mode are restored if possible. If no *filename* argument is given, then the root directory is extracted, which results in the entire content of the tape being extracted, unless the `h` key has been specified.

`t` List the names of the specified files if they occur on the tape. If no file argument is given, then the root directory is listed, which results in the entire content of the tape being listed, unless the `h` key has been specified. The `t` key replaces the function of the old `dumpdir` program.

`i` Interactively restore 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.

Commands are given below. When *dir* or *file* is an argument, the default is the current directory.

`ls` [*dir*] – List the *dir* directory. Entries that are directories are appended with a slash (/). Entries that have been marked for extraction are prepended

with an asterisk (*). If the verbose key is set, each entry's inode number is also listed.

`cd` *dir* – Change the current working directory to *dir*.

`pwd` – Print the full pathname of the current working directory.

`add` [*file*] – Add directory or data file *file* 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 the `h` key was specified on the command line). Files that are on the extraction list are prepended with an asterisk when they are listed by `ls`.

`delete` [*file*] – The current directory or specified argument is deleted from the list of files to be extracted. If a directory is specified, then it and all its descendents are deleted from the extraction list (unless the `h` key was specified on the command line). The most expedient way to extract most of the files from a directory is to add the directory to the extraction list and then delete those files that are not needed.

`extract` – Extract from the dump tape all the files on the extraction list. `Restore` asks you which volume you wish to mount. The fastest way to extract a few files is to start with the last volume and work toward the first volume. To extract files, you need to use "add file" to add the file to the list that `extract` will use.

`setmodes` – All the directories that have been added to the extraction list have their owner, modes, and times set; nothing is extracted from the tape. This is useful for cleaning up after a restore has been prematurely aborted.

`verbose` – Toggle verbose mode (see the `v` key). In verbose mode, the `ls` command lists the inode numbers of all entries, and `restore` prints out information about each file as it is extracted.

`help` – List a summary of the available commands.

`quit` – Exit immediately, even if the extraction list is not empty.

`x` – Exit immediately, even if the extraction list is not empty.

The optional keyletters are:

`b` Use blocking factor *factor*, which is the number of 1024-byte blocks to use per tape record. It must match the blocking factor used to dump the tape. Ideally, this will be the optimal blocking factor for the device you're using. If this keyletter is not used, `restore` tries to determine the tape block size dynamically. See dump(1M) and dump2(1M).

`v` Enter verbose mode. Normally `restore` does its work silently. In verbose mode, `restore` reports the file type and name of each file on which it acts.

`f` Use the next argument to `restore` as the name of the archive instead of `/dev/rmt/0`. If the next argument is '–', `restore` reads from standard input. Thus, `dump` or `dump2` and `restore` can be used in a pipeline to dump and restore a file system with the command

```
dump 0f - /usr | (cd /mnt; restore xf -)
```

If you have DG TCP/IP (DG/UX), you can restore from a remote device. For example,

```
restore rf sys:/dev/rmt/0
```

lets you restore the contents from the tape device "0" on the system "sys" into the current directory. To do this, you must be logged in as root on your own system, and your system must have an entry in the remote host's `/.rhosts` file.

`y` Do not not ask whether the restoration should abort, if a tape error occurs. `Restore` skips over the bad tape block(s) and continues.

`m` Extract by inode numbers rather than by filename. This is useful if only a few files are being extracted and you want to avoid regenerating the complete pathname to the file.

`h` Extract the actual directory rather than the files that it contains. This prevents hierarchical restoration of complete subtrees from the tape.

`s` The next argument to `restore` is a number which selects the file on a multi-file dump tape. File numbering starts at 1.

`Restore`, `dump`(1M), and `dump2`(1M) support symbolic links and control point directories.

FILES

`/dev/rmt/0`	Default tape drive for restoration tapes
`/tmp/rstdir*`	File containing directories on the tape
`/tmp/rstmode*`	File containing owner, mode, and time stamps for directories being restored
`./restoresymtable`	File containing information passed between incremental restorations

DIAGNOSTICS

A bad key character produces an error message.

A read error produces a message. If `y` has been specified or you respond 'y', `restore` attempts to continue restoration.

If the dump extends over more than one tape, `restore` asks you to change tapes. If the `x` or `i` key has been specified, `restore` also asks which volume you wish to mount.

`Restore` performs numerous consistency checks that can produce diagnostic messages. Most messages are self-explanatory or rarely occur. Common error messages are:

Converting to new file system format
: A dump tape created from the old file system has been loaded. It is automatically converted to the new file system format.

filename: not found on tape
: The specified filename was listed in the tape directory but was not found on the tape. This error 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 error can occur when

using a dump tape created on an active file system.

Incremental tape too low
: When doing incremental restore, a tape that was written before the previous incremental tape, or that has too low an incremental level has been loaded.

Incremental tape too high
: When doing incremental restore, a tape that does not begin its coverage where the previous incremental tape left off, or that has too high an incremental level has been loaded.

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 filename is specified, then its contents are probably partially wrong. If an inode is being skipped or the tape is trying to resynchronize, then no extracted files have been corrupted, though 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.

invalid blocking factor, *num*
: See explanation for `b` option.

invalid memory buffer specified, *num*
: See explanation for `g` option.

SEE ALSO

`dump`(1M), `dump2`(1M), `mkfs`(1M), `mount`(1M), `hosts.equiv`(4).

NOTES

`Restore` may give incorrect results when doing incremental restores from dump tapes that were made on active file systems.

A level zero dump must be done after a full restore. Because `restore` runs in user code, it has no control over inode allocation; thus a full dump must be done to get a new set of directories reflecting the new inode numbering, even though the contents of the files are unchanged.

`Restore` complains about socket files (file mode 0140000); it should ignore these files.

When restoring an archive from a medium which is part of a multiple-archive, multiple-medium backup set, `restore` assumes that the first volume is the medium on which this archive begins, regardless of the medium's position in the set. If the archive spans more than one medium, `restore`'s second volume refers to the next medium in the set, and so on.

NAME

rexd – RPC-based remote execution server

SYNOPSIS

/usr/etc/rpc.rexd

DESCRIPTION

rexd is the RPC server (daemon) for remote program execution. rexd is started by inetd(1M) whenever a remote execution request is made.

For noninteractive programs, the standard file descriptors are connected directly to TCP connections. Interactive programs involve pseudo-terminals, in a fashion that is similar to the login sessions provided by rlogin(1). rexd may use NFS to mount file systems specified in the remote execution request.

FILES

/dev/ttyp*n*	pseudo-terminals used for interactive mode
/etc/passwd	authorized users
/etc/hosts.equiv	list of trusted hosts
/tmp_rex/rexd??????	temporary mount points for remote file systems.

DIAGNOSTICS

Diagnostic messages are normally printed on the console, and returned to the requestor.

SEE ALSO

inetd(1M), on(1C), rex(3R), exports(4), hosts.equiv(4M), inetd.conf(4M).

RESTRICTIONS

The rexd server uses the simple trusted host authentication that rlogin and remsh use. For details, see hosts.equiv(4M)

Root cannot execute commands using rexd client programs such as on(1C).

NAME

rexecd – Remote execution server

SYNOPSIS

/usr/bin/rexecd

DESCRIPTION

The rexecd server is for the rexec(3X) routine. The server provides remote execution facilities with authentication based on usernames and encrypted passwords.

The rexecd server is invoked by the inetd server when an incoming connection is detected on the port specified in /etc/services. See inetd(1M) and services(1M) for details. When a service request is received, inetd invokes rexecd and the following protocol is initiated:

1) The server reads characters from the socket up to a null ('\0') byte. The resultant string is interpreted as an ASCII number, base 10.

2) If the number received in step 1 is nonzero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine.

3) A null-terminated username of at most 16 characters is retrieved on the initial socket.

4) A null-terminated encrypted password of at most 16 characters is retrieved on the initial socket.

5) A null-terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.

6) The rexecd server then validates the user as is done at log-in time and, if the authentication was successful, changes to the user's home directory and establishes the user and group protections of the user. If any of these steps fails, the connection is aborted and a diagnostic message is returned.

7) A null byte is returned on the connection associated with the stderr, and the command line is passed to the normal log-in shell of the user. The shell inherits the network connections established by rexecd.

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the stderr, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 above upon successful completion of all the steps prior to the command execution).

username too long	The name is longer than 16 characters.
password too long	The password is longer than 16 characters.
command too long	The command line passed exceeds the size of the argument list (as configured into the system).
Login incorrect	No password file entry exists for the username.
Password incorrect	The wrong password was supplied.
No remote directory	The chdir command to the home directory failed.
Try again	A fork by the server failed.
/bin/sh: ...	The user's log-in shell could not be started.

SEE ALSO

inetd(1M), rexec(3X).

BUGS

Indicating `Login incorrect` instead of `Password incorrect` is a security breach that allows people to probe a system for users with null passwords.

NAME

rlogind - remote login server

SYNOPSIS

rlogind [-d]

DESCRIPTION

The rlogind server is for the rlogin(1C) program. The server provides a remote login facility with authentication based on privileged port numbers. The -d option turns on debugging, with output going to /tmp/rlogind*.

The rlogind program is invoked by the inetd server when an incoming connection is detected on the port specified in /etc/services. See inetd(1M) and services(4) for details. When a service request is received, inetd invokes rlogind and the following protocol is initiated:

1) The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection.

2) The server checks the client's source address. If the address is associated with a host for which no corresponding entry exists in the hostname database (see hosts(4)), the server aborts the connection.

After the source port and address have been checked, rlogind allocates a pseudoterminal (see pty(7)) and manipulates file descriptors so that the slave half of the pseudoterminal becomes the stdin, stdout, and stderr for a login process. The login process is an instance of the login(1) program. The login process may prompt for a password if the remote user is not a trusted user.

The parent of the login process manipulates the master side of the pseudoterminal, operating as an intermediary between the login process and the client instance of the rlogin program. In normal operation, the packet protocol described in pty(7) is invoked to provide ^S/^Q type facilities and propagate interrupt signals to the remote programs.

The rlogin command and rlogind server allow for the dynamic exchange of window size information. This is particularly useful in an environment in which you use windowing software such as X windows. Suppose that within a window, you use rlogin to log in to a host. If you change that window's dimensions through the mouse, the new dimensions are propagated to the corresponding remote server, rlogind. The remote kernel data structures are then changed to reflect these size changes. This information exchange is transparent to a user. For this enhancement to be fully realized, both the local and remote machines must be running the appropriate versions of rlogin and rlogind.

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the stderr, after which any network connections are closed. An error is indicated by a leading byte with a value of one (1).

`Host name for your address (`*client_IP_address*`) unknown`
No entry in the hostname database exists for the client's machine.

`Try again` A fork by the server failed.

`/bin/sh: ...` The user's login shell could not be started.

SEE ALSO

rlogin(1C), inetd(1M), ruserok(3X), services(4), hosts(4), hosts.equiv(4), inetd.conf(4), pty(7).

BUGS

The authentication procedure used here assumes the integrity of each client machine and of the connecting medium. This is not secure but is useful in an "open" environment.

NAME

rmt – start the remote mag tape server

SYNOPSIS

rmt

DESCRIPTION

Rmt is a server process used by the remote dump(1M), dump2(1M), restore(1M), and pmtd(1M) programs in manipulating a magnetic tape drive through an interprocess communication connection. Rmt is normally started up with an rexec(3X) or rcmd(3X) call.

The rmt program accepts requests specific to the manipulation of magnetic tapes, performs the commands, then responds with a status indication. All responses are in ASCII and in one of two forms. Successful commands have responses of:

A*number*<NL>

where *number* is an ASCII representation of a decimal number, and <NL> is the new-line character. Unsuccessful commands are responded to with:

E*errno_val*<NL>*error_message*<NL>

where *errno_val* is one of the possible error numbers described in intro(2), and *error_message* is the corresponding error string as printed from a call to perror(3C).

The protocol consists of the following commands:

O*device*<NL>*mode*<NL>

Open the specified *device* using the indicated *mode*. *device* is a full pathname and *mode* is an ASCII representation of a decimal number suitable for passing to open(2). A successful response number should not be interpreted. If rmt receives additional open commands, the currently open device is closed before the new open is performed.

C*device*<NL>

Close the currently open device. The *device* specified is ignored. A successful response number should not be interpreted.

R*count*<NL>

Read *count* bytes of data from the open device. Rmt performs the requested read(2) and responds with the value returned from the read(2) call if the read was successful; otherwise an error in the standard format is returned. The data read is sent immediately after the response if the read was successful.

W*count*<NL>

Write data onto the open device. Rmt reads *count* bytes from the connection, aborting if a premature EOF is encountered, and writes that data to the open device. The response value is that returned from the write(2) call.

L*whence*<NL>*offset*<NL>

Perform an lseek(2) operation using the specified parameters. The response value is that returned from the lseek(2) call.

I*operation*<NL>*count*<NL>

Perform a MTIOCOP ioctl(2) command using the specified parameters. The parameters are interpreted as the ASCII representations of the decimal values to place in the mt_op and mt_count fields of the structure used in the ioctl call. The return value is the *count* parameter when the operation is successful.

S<NL>

Return the status of the open device, as obtained with a MTIOCGET ioctl(2) call. If the operation was successful, the size of the status buffer and then the status buffer contents are sent. Interpretation of the status buffer contents is implementation specific.

Any other command causes rmt to exit(3C).

Rmt reads requests from standard input and writes responses to standard output.

SEE ALSO

pmtd(1M), dump(1M), dump2(1M), restore(1M), intro(2), ioctl(2), lseek(2), open(2), read(2), write(2), exit(3C), perror(3C), rcmd(3X), rexec(3X), mtio(4).

NOTES

Use rmt(1M) for remote tape access only, not for remote file access. Different operating systems and different hardware may perform device I/O in different ways. In particular lseek, ioctl, and status requests may operate differently. Systems may differ so much that these operations are no longer functionally the same on different machines.

NAME

route – Manipulate the routing tables

SYNOPSIS

route [-f] *command* [net|host] *dest gateway* [*metric*]

DESCRIPTION

Use the route program to manipulate the network routing tables. Use netstat -r to display the routing tables. The route program accepts the following two *commands*:

add: Add a route

delete: Delete a route

The destination named *dest* is a host or network for which the route is "to." A network address should be specified as a complete 4 part Internet address. For example, if the network address is 128.220.3 the network argument should be provided as 128.220.3.0. *gateway* is the gateway to which packets should be addressed. All symbolic names specified for a *dest* or *gateway* are looked up first in the hostname database, hosts(4). If this lookup fails, the name is looked up in the network name database, networks(4).

metric is an option indicating the number of hops to *dest*. If you do not specify a *metric*, route assumes a value of zero (0). If route is to a destination connected via a gateway, the *metric* should be greater than zero (0).

Distinguish routes to a particular host from those to a network by the optional net or host parameter on the command line. If this parameter is absent, routes are distinguished by interpreting the Internet address associated with *dest*. If the destination named *dest* has a local address part of INADDR_ANY, then route is assumed to be to a network; otherwise, it is presumed to be a route to a host.

It is impossible to identify the local address part of an address on a subnetted network, so the net or host parameter must be supplied if you use subnets.

Route uses a raw socket and the SIOCADDRT and SIOCDELRT ioctl's to do its work. Only the superuser may modify the routing tables.

If you specify the -f option, route will "flush" the routing tables of all gateway entries. If you use this with one of the commands described above, the tables are flushed prior to the command's application.

DIAGNOSTICS

add *destination*: gateway *gate_host*, flags *flag_values*

Here, the specified route is being added to the tables. The values printed are from the routing table entry supplied in the ioctl call.

delete *destination*: gateway *gate_host*, flags *flag_values*

Here, the same action (as in the first example) takes place, but when deleting an entry.

destination gate_host done

When you specify the -f flag, each routing table entry deleted is indicated with a

message of this form.

`not in table`

Here, a delete operation was attempted for an entry which wasn't present in the tables.

`routing table overflow`

Here, an `add` operation was attempted, but the system was low on resources and was unable to allocate memory to create the new entry. Clean out the routes that you do not need and try again. The meanings of flag values are provided in the `routing`(6) man page.

SEE ALSO

`netstat`(1C), `inet`(3N), `intro`(6), `routing`(6).

NAME

`routed` – network routing server

SYNOPSIS

`/usr/bin/routed [ -d ] [ -g ] [ -s ] [ -q ] [ -t ] [ *logfile* ]`

DESCRIPTION

Invoke the `routed` server (daemon) to manage the network routing tables. The routing server uses a variant of the Xerox NS Routing Information Protocol in maintaining up to date kernel routing table entries. It is used as a generalized protocol capable of use with multiple address types, but is currently used only for Internet routing within a cluster of networks.

In normal operation `routed` listens on the `udp`(6P) socket for the `route` service (see `services`(4)) for routing information packets. If the host is an internetwork router, it periodically supplies copies of its routing tables to any directly connected hosts and networks.

When `routed` is started, it uses the SIOCGIFCONF `ioctl` to find those directly connected interfaces configured into the system and marked "up" (the software loopback interface is ignored). If multiple interfaces are present, it is assumed that the host will forward packets between networks. The `routed` server then transmits a *request* packet on each interface (using a broadcast packet if the interface supports it) and enters a loop, listening for *request* and *response* packets from other hosts.

When a *request* packet is received, `routed` formulates a reply based on the information maintained in its internal tables. The *response* packet generated contains a list of known routes, each marked with a "hop count" metric (a count of 16, or greater, is considered "infinite"). The metric associated with each route returned provides a metric relative to the sender.

Response packets received by `routed` are used to update the routing tables if one of the following conditions is satisfied:

(1) No routing table entry exists for the destination network or host, and the metric indicates the destination is "reachable" (that is, the hop count is not infinite).

(2) The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed.

(3) The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost effective as the current route.

(4) The new route describes a shorter route to the destination than the one currently stored in the routing tables; the metric of the new route is compared against the one stored in the table to decide this.

When an update is applied, `routed` records the change in its internal tables and updates the kernel routing table. The change is reflected in the next *response* packet sent.

In addition to processing incoming packets, `routed` also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to ensure the invalidation is propagated throughout the local internet.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks. The response is sent to the broadcast address on nets capable of that function, to the destination address on point-to-point links, and to the router's own address on other networks. The normal routing tables are bypassed when sending gratuitous responses. The reception of responses on each network is used to determine that the network and interface are functioning correctly. If no response is received on an interface, another route may be chosen to route around the interface, or the route may be dropped if no alternative is available.

The `routed` command supports several options:

`-d` Enable additional debugging information to be logged, such as bad packets received.

`-g` This flag is used on internetwork routers to offer a route to the "default" destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers.

`-s` Supplying this option forces `routed` to supply routing information whether it is acting as an internetwork router or not. This is the default if multiple network interfaces are present, or if a point-to-point link is in use.

`-q` This is the opposite of the `-s` option.

`-t` If the `-t` option is specified, all packets sent or received are printed on the standard output. In addition, `routed` will not divorce itself from the controlling terminal so that interrupts from the keyboard will kill the process.

Any other argument supplied is interpreted as the name of file in which `routed`'s actions should be logged. This log contains information about any changes to the routing tables and, if not tracing all packets, a history of recent messages sent and received which are related to the changed route.

In addition to the facilities described above, `routed` supports the notion of "distant" *passive* and *active* gateways. When `routed` is started up, it reads the file `/etc/gateways` to find gateways which may not be located using only information from the SIOGIFCONF `ioctl`. See `gateways`(4) for details.

NOTE:

If you use `routed` to define routes, you must specify routes in `/etc/gateways`, and not in `/etc/tcpip.params`. Route definitions specified in `/etc/tcpip.params` are not used by `routed`. Also, if you use `routed`, do not change or implement routes with the `route`(1M) command.

FILES

`/etc/gateways`

SEE ALSO

`gateways`(4), `udp`(6P).

BUGS

The kernel's routing tables may not correspond to those of `routed` when redirects change or add routes. The `routed` server should listen to intelligent interfaces, such as an IMP, and to error protocols, such as ICMP, to gather more information. It does not always detect unidirectional failures in network interfaces (for example, when the output side fails).

The `routed` server should incorporate other routing protocols, such as Xerox NS and EGP. Using separate processes for each requires configuration options to avoid redundant or competing routes.

NAME

rpcinfo - report RPC information

SYNOPSIS

```
/etc/rpcinfo -p [ host ]
/etc/rpcinfo [ -n portnum ] -u host program [ version ]
/etc/rpcinfo [ -n portnum ] -t host program [ version ]
/etc/rpcinfo -b program version
/etc/rpcinfo -d program version
```

DESCRIPTION

The rpcinfo command, which is located in /etc, makes an RPC call to an RPC server and reports what it finds.

OPTIONS

-p Probe the portmapper on *host*, and print a list of all registered RPC programs. If *host* is not specified, it defaults to the value returned by hostname(1).

-u Make an RPC call to procedure 0 of *program* on the specified *host* using UDP, and report whether a response was received.

-t Make an RPC call to procedure 0 of *program* on the specified *host* using TCP, and report whether a response was received.

-n Use *portnum* as the port number for the *−t* and *−u* options instead of the port number given by the portmapper.

-b Make an RPC broadcast to procedure 0 of the specified *program* and *version* using UDP and report all hosts that respond.

-d Delete registration for the RPC service of the specified *program* and *version*. This option can be exercised only by the super-user.

The *program* argument can be either a name or a number.

If a *version* is specified, rpcinfo attempts to call that version of the specified *program*. Otherwise, rpcinfo attempts to find all the registered version numbers for the specified *program* by calling version 0 (which is presumed not to exist; if it does exist, rpcinfo attempts to obtain this information by calling an extremely high version number instead) and attempts to call each registered version. Note: the version number is required for -b and -d options.

EXAMPLES

To show all of the RPC services registered on the local machine use:

```
example% /etc/rpcinfo -p
```

To show all of the RPC services registered on the machine named klaxon use:

```
example% /etc/rpcinfo -p klaxon
```

To show all machines on the local net that are running the Network Information Service service use:

```
example% /etc/rpcinfo -b ypserv 'version' | uniq
```

where 'version' is the current Network Information Service version obtained from the results of the -p switch above.

To delete the registration for version 1 of the walld service use:

```
example% /etc/rpcinfo -d walld 1
```

SEE ALSO

portmap(1M), rpc(4).

Managing ONC/NFS® and Its Facilities on the DG/UX™ System

BUGS

In releases prior to NFSSRC 3.0, the Network File System (NFS) did not register itself with the portmapper; `rpcinfo` cannot be used to make RPC calls to the NFS server on hosts running such releases.

NAME

rshd – Remote shell server

SYNOPSIS

/usr/bin/rshd

DESCRIPTION

The rshd server is for the rcmd(3) routine and, consequently, for the rsh(1) program. The server provides remote execution facilities with authentication based on privileged port numbers.

The rshd server is invoked by the inetd server when an incoming connection is detected on the port specified in /etc/services. See inetd(1M) and services(4) for details. When a service request is received, inetd invokes rshd and the following protocol is initiated:

1) The server checks the client's source port. If the port is not in the range 0-1023, the server aborts the connection.

2) The server reads characters from the socket up to a null ('\0') byte. The resultant string is interpreted as an ASCII number, base 10.

3) If the number received in step 2 is nonzero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine. The source port of this second connection is also in the range 0-1023.

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 hostname database (see hosts(4)), the server aborts the connection.

5) A null-terminated username of at most 16 characters is retrieved on the initial socket. This username is interpreted as a user identity to use on the server's machine.

6) A null-terminated username of at most 16 characters is retrieved on the initial socket. This username 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 length of the command is limited by the upper bound on the size of the system's argument list.

8) The rshd server then validates the user according to the following steps.

a) The username on the client machine is looked up in the password file and a chdir is performed to the user's home directory. If either the lookup or chdir fail, the connection is terminated.

b) If the user is not the superuser, (user id 0), the file /etc/hosts.equiv is consulted for a list of hosts considered equivalent. If the client's hostname is present in this file, the authentication is considered successful.

c) If the lookup fails, or the user is the superuser, then the file .rhosts 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.

9) A null byte is returned on the connection associated with the stderr, and the command line is passed to the normal log-in shell of the user. The shell

inherits the network connections established by `rshd`.

ENVIRONMENT

When you use `rsh` *hostname command*, the environment for the command is not the same as for the login shell. For example, `/etc/TIMEZONE`, `/etc/profile` and `.profile` are not executed for Bourne shell users and `/etc/TIMEZONE.csh`, `/etc/login`, and `.login` are not executed for C shell users (however, `.cshrc` is executed). On the other hand, when you use `rsh` *hostname*, you are performing the equivalent of `rlogin` *hostname*, and the environment is the same as for the login shell.

DIAGNOSTICS

All diagnostic messages are returned on the connection associated with the `stderr`, after which any network connections are closed. An error is indicated by a leading byte with a value of one (1) (zero is returned in step 9 above after successful completion of all the steps prior to 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`	No entry in the hostname database exists for the client's machine.
`Login incorrect`	No password file entry exists for the username.
`No remote directory`	The `chdir` command to the home directory failed.
`Permission denied`	The authentication procedure (described above) failed.
`Can't make pipe`	The pipe needed for the `stderr` wasn't created.
`Try again`	A `fork` by the server failed.
`/bin/sh: ...`	The user's login shell could not be started.

SEE ALSO

`rsh`(1), `inetd`(1M), `rcmd`(3), `hosts.equiv`(4)

BUGS

The authentication procedure used here assumes the integrity of each client machine and of the connecting medium. This is not secure but is useful in an "open" environment.

NAME

rstatd – kernel statistics server

SYNOPSIS

/usr/etc/rpc.rstatd

DESCRIPTION

rstatd is a server (daemon) that returns performance statistics obtained from the kernel. The rstatd server is normally invoked by inetd(1M).

SEE ALSO

rup(1C), inetd(1M).

NAME

runacct – run daily accounting

SYNOPSIS

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

where:

mm A two digit integer (01 through 12) indicating month
dd A two-digit integer indicating day of month
state One of the states (SETUP, etc.) described below

DESCRIPTION

Runacct is the main daily accounting shell procedure. It is normally initiated via cron(1M). Runacct processes connect, fee, disk, and process accounting files. It also prepares summary files for prdaily or billing purposes.

Runacct takes care not to damage active accounting files or summary files in the event of errors. It records its progress by writing descriptive diagnostic messages into active. 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. Runacct uses a series of lock files to protect against re-invocation. The files lock and lock1 are used to prevent simultaneous invocation, and lastdate is used to prevent more than one invocation per day.

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

SETUP	Move active accounting files into working files.
WTMPFIX	Verify integrity of wtmp file, correcting date changes if necessary.
CONNECT1	Produce connect session records in ctmp.h format.
CONNECT2	Convert ctmp.h records into tacct.h format.
PROCESS	Convert process accounting records into tacct.h format.
MERGE	Merge the connect and process accounting records.
FEES	Convert output of *chargefee* into tacct.h format and merge with connect and process accounting records.
DISK	Merge disk accounting records with connect, process, and fee accounting records.
MERGETACCT	Merge the daily total accounting records in daytacct with the summary total accounting records in /usr/adm/acct/sum/tacct.
CMS	Produce command summaries.
USEREXIT	Any installation-dependent accounting programs can be included here.
CLEANUP	Cleanup temporary files and exit.

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 . The lock files and lastdate file must be removed before runacct can be restarted. The argument *mmdd* is necessary if runacct is being restarted, and specifies the month and day for which runacct will rerun the accounting. Entry point for processing is based on the contents of statefile; to override this, include the desired *state* on the command line

to designate where processing should begin.

EXAMPLES

To start `runacct`.

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

To restart `runacct`.

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

To restart `runacct` at a specific *state*.

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

FILES

`/etc/wtmp`
`/usr/adm/pacct*`
`/usr/src/cmd/acct/tacct.h`
`/usr/src/cmd/acct/ctmp.h`
`/usr/adm/acct/nite/active`
`/usr/adm/acct/nite/daytacct`
`/usr/adm/acct/nite/lock`
`/usr/adm/acct/nite/lock1`
`/usr/adm/acct/nite/lastdate`
`/usr/adm/acct/nite/statefile`
`/usr/adm/acct/nite/ptacct*.`*mmdd*

SEE ALSO

`acct`(1M), `acctcms`(1M), `acctcom`(1), `acctcon`(1M), `acctmerg`(1M), `acctprc`(1M), `acctsh`(1M), `cron`(1M), `fwtmp`(1M), `mail`(1), `acct`(2), `acct`(4), `utmp`(4), `sysadm`(1M).

BUGS

Normally it is not a good idea to restart `runacct` in the `SETUP` *state*. Run `SETUP` manually and restart via:

```
runacct mmdd WTMPFIX
```

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

NAME

rusersd – network username server

SYNOPSIS

/usr/etc/rpc.rusersd

DESCRIPTION

Rusersd is a server (daemon) that returns a list of users on the network. The rusersd server is normally invoked by inetd(1M).

SEE ALSO

rusers(1C), inetd(1M).

NAME

rwalld – network rwall server

SYNOPSIS

/usr/etc/rpc.rwalld

DESCRIPTION

Rwalld is a server (daemon) that handles rwall(1C) and shutdown(2) requests. It is implemented by calling wall(1) to all the appropriate network machines. The rwalld server is normally invoked by inetd(1M).

SEE ALSO

inetd(1M), rwall(1C), wall(1), shutdown(2).

NAME

rwhod – system status server

SYNOPSIS

/usr/bin/rwhod

DESCRIPTION

The rwhod server maintains the database used by the rwho(1C) and ruptime(1C) programs. Its operation is predicated on the ability to broadcast messages on a network.

The rwhod server operates as both a producer and consumer of status information. As a producer of information, it periodically queries the state of the system and constructs status messages that are broadcast on a network. As a consumer of information, it listens for other rwhod servers' status messages, validating them, then recording them in a collection of files located in the directory /var/spool/rwho.

The rwhod server transmits and receives messages at the port indicated in the rwho service specification in /etc/services. The messages sent and received are of the form:

```
struct      outmp {
      char  out_line[8];/* tty name */
      char  out_name[8];/* user id */
      long  out_time;/* time on */
};

struct      whod {
      char  wd_vers;
      char  wd_type;
      char  wd_pad[2];
      int   wd_sendtime;
      int   wd_recvtime;
      char  wd_hostname[32];
      int   wd_loadav[3];
      int   wd_boottime;
      struct      whoent {
            struct outmp we_utmp;
            int   we_idle;
      } wd_we[100];
};
```

All fields are converted to network byte order prior to transmission. The load averages represent system loads over the 5-, 10-, and 15-minute intervals prior to a server's transmission. The hostname included is that returned by the gethostname(2) system call. The array at the end of the message contains information about the users logged in to the sending machine. This information includes the contents of the utmp(4) entry for each terminal line in use and a value indicating the time since a character was last received on the terminal line.

Messages received by the rwhod server are discarded unless they originated at the rwhod server's port. In addition, if the host's name, as specified in the message, contains any unprintable ASCII characters, the message is discarded. Valid messages received by rwhod are placed in files named whod.hostname in the directory /var/spool/rwho. These files contain only the most recent message, in the format described above.

Status messages are generated approximately once every three minutes. A system is considered down if no messages are received from it for 11 minutes.

NOTE: The broadcast time of three minutes applies to hosts running 4.3 BSD compatible rwhod. Hosts running 4.2 BSD compatible rwhod will broadcast once a minute.

SEE ALSO

`ruptime`(1C), `rwho`(1C).

BUGS

This command should relay status information between networks. People often interpret the server dying as a machine going down.

NAME

sac – service access controller

SYNOPSIS

sac -t *sanity_interval*

DESCRIPTION

The Service Access Controller (SAC) is the overseer of the server machine. It is started when the server machine enters multiuser mode. The SAC performs several important functions as explained below.

Customizing the SAC environment. When sac is invoked, it first looks for the per-system configuration script /etc/saf/_sysconfig. sac interprets _sysconfig to customize its own environment. The modifications made to the SAC environment by _sysconfig are inherited by all the children of the SAC. This inherited environment may be modified by the children.

Starting port monitors. After it has interpreted the _sysconfig file, the sac reads its administrative file /etc/saf/_sactab. _sactab specifies which port monitors are to be started. For each port monitor to be started, sac forks a child [fork(2)] and creates a utmp entry with the *type* field set to LOGIN_PROCESS. Each child then interprets its per-port monitor configuration script /etc/saf/*pmtag*/_config, if the file exists. These modifications to the environment affect the port monitor and will be inherited by all its children. Finally, the child process execs the port monitor, using the command found in the _sactab entry. (See sacadm; this is the command given with the -c option when the port monitor is added to the system.)

Polling port monitors to detect failure. The -t option sets the frequency with which sac polls the port monitors on the system. This time may also be thought of as half of the maximum latency required to detect that a port monitor has failed and that recovery action is necessary.

Administrative functions. The Service Access Controller represents the administrative point of control for port monitors. Its administrative tasks are explained below.

When queried (sacadm with either -l or -L), the Service Access Controller returns the status of the port monitors specified, which sacadm prints on the standard output. A port monitor may be in one of six states:

ENABLED — The port monitor is currently running and is accepting connections. See sacadm(1M) with the -e option.

DISABLED — The port monitor is currently running and is not accepting connections. See sacadm with the -d option, and see NOTRUNNING, below.

STARTING — The port monitor is in the process of starting up. STARTING is an intermediate state on the way to ENABLED or DISABLED.

FAILED — The port monitor was unable to start and remain running.

STOPPING — The port monitor has been manually terminated but has not completed its shutdown procedure. STOPPING is an intermediate state on the way to NOTRUNNING.

NOTRUNNING — The port monitor is not currently running. (See sacadm with -k.) This is the normal "not running" state. When a port monitor is killed, all ports it was monitoring are inaccessible. It is not possible for an external user to tell whether a port is not being monitored or the system is down. If the port monitor is not killed but is in the

DISABLED state, it may be possible (depending on the port monitor being used) to write a message on the inaccessible port telling the user who is trying to access the port that it is disabled. This is the advantage of having a DISABLED state as well as the NOTRUNNING state.

When a port monitor terminates, the SAC removes the utmp entry for that port monitor.

The SAC receives all requests to enable, disable, start, or stop port monitors and takes the appropriate action.

The SAC is responsible for restarting port monitors that terminate. Whether or not the SAC will restart a given port monitor depends on two things:

- the restart count specified for the port monitor when the port monitor was added by sacadm; this information is included in /etc/saf/*pmtag*/_sactab
- the number of times the port monitor has already been restarted

FILES

/etc/saf/_sactab
/etc/saf/_sysconfig
/var/adm/utmp
/var/saf/_log

SEE ALSO

sacadm(1M), pmadm(1M).

NAME

sacadm - service access controller administration

SYNOPSIS

```
sacadm -a -p pmtag -t type -c cmd -v ver [-f dx] [-n count] \
   [-y comment] [-z script]

sacadm -r -p pmtag

sacadm -s -p pmtag

sacadm -k -p pmtag

sacadm -e -p pmtag

sacadm -d -p pmtag

sacadm -l [-p pmtag | -t type]

sacadm -L [-p pmtag | -t type]

sacadm -g -p pmtag [-z script]

sacadm -G [-z script]

sacadm -x [-p pmtag]
```

DESCRIPTION

sacadm is the administrative command for the upper level of the Service Access Facility hierarchy, that is, for port monitor administration. sacadm performs the following functions:

- adds or removes a port monitor
- starts or stops a port monitor
- enables or disables a port monitor
- installs or replaces a per-system configuration script
- installs or replaces a per-port monitor configuration script
- prints requested port monitor information

Requests about the status of port monitors (-l and -L) and requests to print per-port monitor and per-system configuration scripts (-g and -G without the -z option) may be executed by any user on the system. Other sacadm commands may be executed only by a privileged user.

The options have the following meanings:

-a Add a port monitor. When adding a port monitor, sacadm creates the supporting directory structure in /etc/saf and /var/saf and adds an entry for the new port monitor to /etc/saf/_sactab. The file _sactab already exists on the delivered system. Initially, it is empty except for a single line, which contains the version number of the Service Access Controller.

Unless the command line that adds the new port monitor includes a -f option with the argument x, the new port monitor will be started. Because of the complexity of the options and arguments that follow the -a option, it may be convenient to use a command script or the menu system to add port monitors. If you use the menu system, enter sysadm ports and then choose the port_monitors option.

-c *cmd*

Execute the command string *cmd* to start a port monitor. The -c option may

be used only with a `-a`. A `-a` option requires a `-c`.

`-d` Disable the port monitor *pmtag*.

`-e` Enable the port monitor *pmtag*.

`-f dx` The `-f` option specifies one or both of the following two flags which are then included in the flags field of the `_sactab` entry for the new port monitor. If the `-f` option is not included on the command line, no flags are set and the default conditions prevail. By default, a port monitor is started. A `-f` option with no following argument is illegal.

- `d` Do not enable the new port monitor.
- `x` Do not start the new port monitor.

`-g` The `-g` option is used to request output or to install or replace the per-port monitor configuration script `/etc/saf/`*pmtag*`/_config`. `-g` requires a `-p` option. The `-g` option with only a `-p` option prints the per-port monitor configuration script for port monitor *pmtag*. The `-g` option with a `-p` option and a `-z` option installs the file `script` as the per-port monitor configuration script for port monitor *pmtag*. Other combinations of options with `-g` are invalid.

`-G` The `-G` option is used to request output or to install or replace the per-system configuration script `/etc/saf/_sysconfig`. The `-G` option by itself prints the per-system configuration script. The `-G` option in combination with a `-z` option installs the file `script` as the per-system configuration script. Other combinations of options with a `-G` option are invalid.

`-k` Stop port monitor *pmtag*.

`-l` The `-l` option is used to request port monitor information. The `-l` by itself lists all port monitors on the system. The `-l` option in combination with the `-p` option lists only the port monitor specified by *pmtag*. A `-l` in combination with the `-t` option lists all port monitors of type *type*. Any other combination of options with the `-l` option is invalid.

`-L` The `-L` option is identical to the `-l` option except that the output appears in a condensed format.

`-n` *count*
Set the restart count to *count*. If a restart count is not specified, count is set to `0`. A count of `0` indicates that the port monitor is not to be restarted if it fails.

`-p` *pmtag*
Specifies the tag associated with a port monitor.

`-r` Remove port monitor *pmtag*. `sacadm` removes the port monitor entry from `/etc/saf/_sactab`. If the removed port monitor is not running, then no further action is taken. If the removed port monitor is running, the Service Access Controller (SAC) sends it `SIGTERM` to indicate that it should shut down. Note that the port monitor's directory structure remains intact.

`-s` Start a port monitor. The SAC starts the port monitor *pmtag*.

`-t` *type*
Specifies the port monitor type.

`-v` *ver* Specifies the version number of the port monitor. This version number may be given as

`-v` `` `pmspec -V` ``

where *pmspec* is the special administrative command for port monitor *pmtag*. This special command is `ttyadm` for `ttymon` and `nlsadmin` for `listen`. The version stamp of the port monitor is known by the command and is returned when *pmspec* is invoked with a `-V` option.

`-x` The `-x` option by itself tells the SAC to read its database file (`_sactab`). The `-x` option with the `-p` option tells port monitor *pmtag* to read its administrative file.

`-y` *comment*
Include *comment* in the `_sactab` entry for port monitor *pmtag*.

`-z script`
Used with the `-g` and `-G` options to specify the name of a file that contains a configuration script. With the `-g` option, `script` is a per-port monitor configuration script; with `-G` it is a per-system configuration script. Modifying a configuration script is a three-step procedure. First a copy of the existing script is made (`-g` or `-G`). Then the copy is edited. Finally, the copy is put in place over the existing script (`-g` or `-G` with `-z`).

OUTPUT

If successful, `sacadm` will exit with a status of `0`. If `sacadm` fails for any reason, it will exit with a nonzero status. Options that request information will write the information on the standard output. In the condensed format (`-L`), port monitor information is printed as a sequence of colon-separated fields; empty fields are indicated by two successive colons. The standard format (`-l`) prints a header identifying the columns, and port monitor information is aligned under the appropriate headings. In this format, an empty field is indicated by a hyphen. The comment character is `#`.

EXAMPLES

The following command line adds a port monitor. The port monitor tag is `npack`; its type is `listen`; if necessary, it will restart three times before failing; its administrative command is `nlsadmin`; and the configuration script to be read is in the file `script`:

```
sacadm -a -p npack -t listen -c /usr/lib/saf/listen npack \
   -v `nlsadmin -V` -n 3 -z script
```

Remove a port monitor whose tag is `pmtag`:

```
sacadm -r -p pmtag
```

Start the port monitor whose tag is `pmtag`:

```
sacadm -s -p pmtag
```

Stop the port monitor whose tag is `pmtag`:

```
sacadm -k -p pmtag
```

Enable the port monitor whose tag is `pmtag`:

```
sacadm -e -p pmtag
```

Disable the port monitor whose tag is `pmtag`:

```
sacadm -d -p pmtag
```

List status information for all port monitors:

```
sacadm -l
```

List status information for the port monitor whose tag is `pmtag`:

```
sacadm -l -p pmtag
```

List the same information in condensed format:

```
sacadm -L -p pmtag
```

List status information for all port monitors whose type is `listen`:

```
sacadm -l -t listen
```

Replace the per-port monitor configuration script associated with the port monitor whose tag is `pmtag` with the contents of the file `file.config`:

```
sacadm -g -p pmtag -z file.config
```

FILES

`/etc/saf/_sactab`
`/etc/saf/_sysconfig`
`/etc/saf/`*`pmtag`*`/_config`

SEE ALSO

`pmadm`(1M), `sac`(1M), `doconfig`(3N).

NAME

sar: sa1, sa2, sadc - system activity report package

SYNOPSIS

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

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

/usr/lib/sa/sa2 [-ubdycwaqvmprA] [-s *time*] [-e *time*] [-i *sec*]

DESCRIPTION

Sar(1M) accesses system activity data automatically on a routine basis. Such data can also be accessed at the special request of a user through sar(1). The operating system contains a number of counters that are incremented as various system actions occur. These include CPU utilization counters, buffer usage counters, disk and tape I/O activity counters, TTY device activity counters, switching and system-call counters, file-access counters, queue activity counters, and counters for inter-process communications.

Sadc and the shell procedures sa1 and sa2 sample, save, and process this data.

Sadc, the data collector, samples system data *n* times every *t* seconds and writes in binary format to *ofile* or to standard output. If *t* and *n* are omitted, a special record is written. This facility is used at system boot time to mark the time at which the counters restart from zero. The /etc/init.d/rc.account entry:

```
su sys -c "/usr/lib/sa/sadc /usr/adm/sa/sa`date +%d`"
```

writes the special record to the daily data file to mark the system restart.

The shell script sa1, a variant of sadc, collects and stores data in binary file /usr/adm/sa/sa*dd* (*dd* is the current day). The arguments *t* and *n* write records *n* times at an interval of *t* seconds; the default is to write once. The following crontab [see cron(1M)] entries will produce records every 20 minutes during working hours and hourly otherwise:

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

The shell script sa2, a variant of sar(1), writes a daily report in file /usr/adm/sa/sar*dd*. The options are explained in sar(1). The crontab entry:

```
5 18 * * 1-5 su adm -c "/usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 3600 -A"
```

reports important activities hourly during the working day.

The binary data file consists of a tblmap structure followed by a series of sa structures, defined as follows:

```
struct tblmap {
      char    sa_magic[4];        /* a magic "number" */
      short   sa_revision;        /* a version id */
      char    sa_sysname[12];/*from monitored system's uname struct*/
      char    sa_nodename[12];
      char    sa_release[24];
      char    sa_version[24];
      char    sa_machine[24];
      int     sa_hertz;           /* monitored system's hertz value */
      char    devnm[NDEVS][25];   /* device names */
} tblmap;
#define  SAMAGIC     "sar"
#define  SAVERSION   0430

struct sysinfo {
      time_t            cpu[5];
#define          CPU_IDLE          0
#define          CPU_USER          1
#define          CPU_KERNEL        2
#define          CPU_WAIT          3
#define          CPU_SXBRK         4
      time_t            wait[3];
#define          W_IO              0
#define          W_SWAP            1
#define          W_PIO             2
      unsigned long     bread;
      unsigned long     bwrite;
      unsigned long     lread;
      unsigned long     lwrite;
      unsigned long     phread;
      unsigned long     phwrite;
      unsigned long     swapin;
      unsigned long     swapout;
      unsigned long     bswapin;
      unsigned long     bswapout;
      unsigned long     pswitch;
      unsigned long     syscall;
      unsigned long     sysread;
      unsigned long     syswrite;
      unsigned long     sysfork;
      unsigned long     sysexec;
      unsigned long     runque;
      unsigned long     runocc;
      unsigned long     swpque;
      unsigned long     swpocc;
      unsigned long     iget;
      unsigned long     namei;
      unsigned long     dirblk;
      unsigned long     readch;
      unsigned long     writech;
      unsigned long     rcvint;
      unsigned long     xmtint;
      unsigned long     mdmint;
```

```
        unsigned long    rawch;
        unsigned long    canch;
        unsigned long    outch;
        unsigned long    msg;
        unsigned long    sema;
        unsigned long    pnpfault;
        unsigned long    wrtfault;
};

struct minfo {
        unsigned long    freemem;
        unsigned long    freeswap;
        unsigned long    vfault;
        unsigned long    pfault;
        unsigned long    file;
        unsigned long    freedpgs;
};

struct dinfo {
        time_t          serve;
};

struct sa {
        struct sysinfo si;       /* system statistics */
        struct minfo   mi;       /* memory and paging statistics */
        struct dinfo   di;       /* (not used) */
        unsigned int   minserve; /* (not used) */
        unsigned int   maxserve; /* (not used) */
        unsigned int   szinode;  /* current entries of inode table */
        unsigned int   szfile;   /* current entries of file table */
        unsigned int   szproc;   /* current entries of proc table */
        unsigned int   szlckf;   /* cur size of file record hdr. table*/
        unsigned int   szlckr;   /* cur size of file record lock table*/
        unsigned int   mszinode; /* max size of inode table */
        unsigned int   mszfile;  /* max size of file table */
        unsigned int   mszproc;  /* max size of proc table */
        unsigned int   mszlckf;  /* max size of file record hdr. table*/
        unsigned int   mszlckr;  /* max size of file record lock table*/
        unsigned long  inodeovf; /* cumul. overflows of inode table */
        unsigned long  fileovf;  /* cumul. overflows of file table */
        unsigned long  procovf;  /* cumul. overflows of proc table */
        time_t         ts;       /* time stamp, seconds */
        int            apstate;  /* number of processors */
        unsigned long  devio[NDEVS][4];/*dev info for up to NDEVS units*/
#define  IO_OPS      0     /* cumul. I/O requests  */
#define  IO_BCNT     1     /* cumul. blocks transferred */
#define  IO_ACT      2     /* cumul. drive active time in ticks  */
#define  IO_RESP     3     /* cumul. I/O resp time in ticks  */
};
```

Note that not all elements of all structures are used by the DG/UX implementation of sar.

FILES

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

SEE ALSO

`cron`(1M), `sar`(1), `timex`(1).

NAME

sendmail, newaliases, smtp, mailq - Internet mail transport service

SYNOPSIS

sendmail [*flags*] [*address* ...]

newaliases

smtp [*flags*]

mailq

DESCRIPTION

The sendmail program sends a message to one or more recipients, routing the message over whatever networks are necessary. The sendmail program does internetwork forwarding, as necessary, to deliver the message to the correct place.

The sendmail program is not intended as a user interface routine; other programs provide user-friendly front ends; sendmail is used only to deliver pre-formatted messages.

With no flags, sendmail reads its standard input up to a ^D or a line with a single dot and sends a copy of the letter found there to all of the addresses listed. It determines the network to use based on the syntax and contents of the addresses.

Local addresses are looked up in a file and aliased appropriately. Aliasing can be prevented by using the -n flag. Normally the sender is not included in any alias expansions, e.g., if 'john' sends to 'group', and 'group' includes 'john' in the expansion, then the letter will not be delivered to 'john'. To override this feature, use the -om option. Recursive alias expansion is automatically suppressed.

When it gets a message, sendmail attempts to reconcile the name of the addressee for any possible alias, unless aliasing is suppressed. It first interrogates the local alias database. If it does not find the alias there, sendmail queries the Network Information Services' (NIS) mail.aliases map. If the name alias does not exist in mail.aliases, sendmail attempts to deliver the mail to the named addressee. If NIS is not installed on the host, or if NIS becomes unavailable, sendmail checks the local alias database as usual. For more information about mail.aliases, see *Managing ONC/NFS and Its Facilities on the DG/UX System*.

Flags are:

-ba Run in ARPANET mode.

-bd Run as a server (daemon). This is the same as saying smtp in the command line.

-bi Initialize the alias database. This is the same as using newaliases on the command line.

-bm Deliver mail in the usual way (default).

-bp Print the contents of the mail queue.

-bs Use the SMTP protocol as described in RFC821.

-bt Run in address test mode. This mode reads addresses and shows the steps in parsing; it is used for debugging configuration tables.

-bv Verify names only - do not try to collect or deliver a message. Verify mode is normally used for validating users or mailing lists.

-bz Create a frozen configuration file.

-C*file* Use alternate configuration file. When given this option, sendmail runs as the invoking user, not root.

-d*X* Set debugging value to `X`. In general, this will cause `sendmail` to print more information about what it is doing. See the sendmail chapter for more information on different ways to set debugging levels.

`-F` Use the full name for the sender on the `From:` line.

-f*name* Sets the name of the "from" person (that is, the sender of the mail). `-f` can be used only by the trusted users, as defined in the configuration file, typically `root`, `daemon`, and `network`.

`-hN` Set the hop count to `N`. The hop count is incremented every time the mail is processed. When it reaches a limit, the mail is returned with an error message, possibly the victim of an aliasing loop.

`-n` Don't do aliasing.

`-ox`*value* Set option `x` to the specified *value*. Options are described below.

`-q[`*time*`]` Process saved messages in the queue at given intervals. (This option requires superuser privilege.) If time is omitted, process the queue once. Time is given as a tagged number and uses the following abbreviations:

`s` = seconds
`m` = minutes
`h` = hours
`d` = days
`w` = weeks

For example, "`-q1h30m`" or "`-q90m`" both set the time interval to 1 hour 30 minutes.

-r*name* Obsolete form of the `-f` flag.

`-t` Use the `To:` and `Cc:` lines of the message to determine where the mail should go.

`-v` Go into verbose mode. Alias expansions will be announced.

-Z*file* Use a different frozen configuration file. When given this option, sendmail runs as the invoking user, not root.

There are also a number of processing options that may be set. Normally, these will be used only by a system administrator. Options may be set either on the command line using the `-o` flag or in the configuration file. The options are:

A*file* Use alternate alias file.

a*[N]* If set, wait `N` minutes for an `@:@` entry to exist in the alias database before rebuilding the database. If it does not appear in `N` minutes, rebuild the database. If `N` is not specified, the wait is 5 minutes.

B*c* Substitute the character *c* for any blank encountered in the address.

`c` On mailers that are considered "expensive" to connect to (as designated by the `-e` mailer flag), don't initiate immediate connection. Messages will be queued.

d*x* Set the delivery mode to *x*. Delivery modes are as follows:

i = interactive (synchronous) delivery
b = background (asynchronous) delivery
q = queue only – i.e., actual delivery is done the next time the queue is run.

D If necessary, try to automatically rebuild the alias database. If this option is not used, the newaliases command must be invoked each time the aliases file is updated.

e*x* Set error processing to mode *x*. Valid modes are as follows:

e = mail errors back and give zero exit status
m = mail back the error message
w = "write" back the error message (or mail it back if the sender is not logged in)
p = print the errors on the terminal (default)
q = throw away error messages

For w and p modes, the text of the message is also appended to the file dead.letter in the sender's home directory. This option works only with interactive delivery by invoking sendmail with the -v flag. While working with non-interactive mode, error messages are always mailed to the sender.

F*mode* The file permission mode to use when creating temporary files.

f Save UNIX-style "From " lines at the front of messages.

g*id* The default group ID to use when calling mailers.

H*file* Specify the SMTP help file.

I If set, sendmail uses the domain name system (DNS) to determine where to route mail on the Internet. Unlike the Berkeley implementation of sendmail, if this option is not set, sendmail, does not attempt to get MX records. If set and the DNS is not available, sendmail queues the mail. Therefore you should never set this option if you are not using the DNS; mail will never get delivered.

i Do not interpret a dot on a line by itself as a message terminator.

K*xname*

or K*x%map*

Declare the keyed database *x* to be associated with the dbm(3X) file named *name*. Always specify *x* as a single letter. If you follow *x* with a percent sign (%), specify the name of a Network Information Services (NIS) *map* after the percent sign.

L*n* The log level.

M*xvalue* Set the macro *x* to *value*. Use this option only from the command line (for example, sendmail -oMDARPA); recall that in the configuration file, you use D*xval* to set macros (for example, DDARPA).

m Mail to the sender, even if it is in alias expansion.

N*netname* Specify the name of the home network. The argument of an SMTP HELO command is checked against *hostname.netname*, where *hostname* is

requested from the kernel for the current connection. If the argument to HELO does not match the name obtained from the kernel, the name obtained from the kernel is added to `Received:` lines to assist in message tracing.

`o` If this option is set, the message may have old style headers. If the option is not set, the message is guaranteed to have new style headers (that is, commas instead of spaces between addresses). If set, an adaptive algorithm is used that will correctly determine the header format in most cases.

`Q`*directory* Select the directory in which to queue messages.

`Q`*factor* The `sendmail` program divides the value of the *factor* you specify by the difference between the current load average and the load average limit to determine the maximum message priority of messages to be sent immediately. When the resulting quotient is less than the priority of the message, the job is queued rather than run immediately.

`r`*time* The timeout on reads; if none is set, `sendmail` will wait forever for a mailer. Unlike other implementations of `sendmail`, this one associates the read timeout with the reception of an entire message. A timeout between 1 and 2 hours is recommended.

`/` If the split rewriting option is set, sender and recipient envelope addresses are processed with rulesets 1 and 2 respectively, and sender and recipient header addresses are processed with rulesets 5 and 6 respectively. (You must define rulesets 5 and 6.) This is contrary to the default, when both envelope and header addresses are processed with rulesets 1 and 2.

`S`*file* Log mail statistics in the specified *file*. The statistics logged are the number of message to and from each mailer and the number of kilobytes transferred to and from each mailer. The default value is `/etc/sendmail.st`. If you use a non-default statistics file, first create an empty file.

`s` Always create the queue file, even under circumstances where it is not strictly necessary.

`T`*time* Set the timeout on messages in the queue to the specified time. After sitting in the queue for this amount of time, they will be returned to the sender. The default is three days.

`u`*id* Set the default user ID for mailers.

`v` Run in verbose mode.

`x`*N* Set the load average above which to queue messages to *N*, where *N* is a real number. The default value for *N* is 2.0. The `sendmail` program will only queue messages to conserve system resources.

`X`*n* Set the load average above which to refuse incoming messages to *n*, where *n* is a real number. The default value for *n* is 3.0. The `sendmail` program will not accept any incoming connections until the load average falls below *n*.

`Y` If set, `sendmail` uses a distinct process to deliver each job that is run from the queue. Use this option if your system has little memory, since otherwise `sendmail` uses considerable memory when processing the queue.

The destination address for sendmail may be a name of a program to pipe the mail to, rather than a username. Such a message must be included in the aliases file and must start with a vertical bar. It may be necessary to quote the name of the user to keep sendmail from suppressing the blanks between arguments.

The sendmail program returns an exit status describing what it did.

```
0     EX_OK             Successful completion on all addresses.
64    EX_USAGE          Incorrect arguments to command.
65    EX_DATAERR        Input data was incorrect.
66    EX_NOINPUT        An input file (not a system file)
                        did not exist or was not readable.
67    EX_NOUSER         Username not recognized.
68    EX_NOHOST         Hostname not recognized.
69    EX_UNAVAILABLE    Necessary resources were not available.
70    EX_SOFTWARE       Software error, including bad arguments.
71    EX_OSERR          Temporary operating system error;
                        for example, cannot fork.
72    EX_OSFILE         Some system file does not exist.
73    EX_CANTCREAT      A user-specified file cannot be created
74    EX_IOERR          An error occurred while reading from or
                        writing to some file.
75    EX_TEMPFAIL       Message not sent immediately; is queued.
76    EX_PROTOCOL       The remote system returned something
                        violating the SMTP protocol.
77    EX_NOPERM         Invoking user lacked permission for
                        requested operation.
78    EX_CONFIG         Configuration error.
```

If you invoke it as newaliases, sendmail will rebuild the alias database. If you invoke it as mailq, sendmail will print the contents of the mail queue. If invoked as smtp, sendmail starts the server.

FILES

Some of these pathnames are specified in /etc/sendmail.cf. Thus, these values are only defaults.

/etc/aliases	Text file for alias database
/etc/aliases.dir	Compiled alias database - contains database index
/etc/aliases.pag	Compiled alias database - contains database data
/etc/mailstats.st	Repository for mail statistics
/etc/sendmail.cf	Configuration file
/etc/sendmail.fc	Frozen version of configuration file
/etc/sendmail.hf	Help file for sendmail
/etc/sendmail.pid	File containing process ID of the smtp server
/usr/bin/newaliases	To initialize aliases
/usr/bin/dbm	Program to build dbm files for sendmail
/usr/bin/mailq	To print the mail queue
/usr/bin/mailstats	Program to print/clear accumulated mail statistics

/usr/bin/smtp	To start sendmail server
/var/spool/mqueue/*	Temp files
/usr/bin/mail	To deliver local mail
/usr/bin/mailx	Interactive message processing system
/etc/mail/mailx.rc	Sendmail to deliver messages (mail is default)
/etc/tcpip.params	TCP/IP parameters
/usr/sbin/init.d/rc.tcpipserv	Must start smtp server
/etc/passwd	Must include entry for local mailer
/etc/services	Must include smtp entry

SEE ALSO

dbm(1), mail(1), mailstats(1), mailx(1), aliases(4).

BUGS

The sendmail program converts blanks in addresses to dots. This is incorrect according to the old ARPANET mail protocol RFC733 (NIC 41952), but is consistent with the new protocols (RFC822).

NAME

setany – set value of SNMP MIB-object

SYNOPSIS

setany *host community* [*object* type *value*] ...

where:

host	A hostname or Internet address
community	A community string
object	An object instance
value	A value to be assigned to *object*

DESCRIPTION

Use the setany command to assign values to object instances. The command sends an SNMP message containing the *community* string and a SetReqest-PDU requesting the *value* of the type specified by type to be assigned to the *object* by the agent running on *host*.

Specify the *host* as either a hostname or an Internet address in dot-notation.

The *community* string is a text string used by the agent to authenticate the request. For the operation to be successful the community must be configured with read-write access for the agent running on *host*.

Specify the *object* as either an object identifier in dot-notation or as an object descriptor using a text string.

Types are:

-i *value* is integer type (e.g., 123)

-o *value* is octet string type in hexadecimal notation (e.g., "FF EE 12")

-s *value* is display string type (e.g., "hello world")

-d *value* is object identifier type in dot notation (e.g., 1.3.6.1.2.1.2)

-a *value* is Internet address type in dot notation (e.g., 127.0.0.1)

-c *value* is counter type (e.g., 12345)

-g *value* is guage type (e.g., 12345)

-t *value* is time-tick type (e.g., 12345)

EXAMPLES

The following example demonstrates how setany can be used to change the state of the loopback device.

```
$ getmany myhost public ifOperStatus

Name: ifOperStatus.1
Value: 1

Name: ifOperStatus.2
Value: 2

$ getone myhost public ifDescr.2

Name: ifDescr.2
Value: loop0

$ ifconfig loop0
```

```
loop0: 127.0.0.1 flags=48<LOOPBACK, RUNNING>
netmask = 0xff000000 metric = 0

$ setany myhost MyReadWriteCommunity ifAdminStatus.2 -i 1

Name: ifAdminStatus.2
Value: 1

$ ifconfig loop0
loop0: 127.0.0.1 flags=449<UP,LOOPBACK,RUNNING,STARTED>
netmask = 0xff000000 metric = 0
```

DIAGNOSTICS

Exit status is 0 upon success.

Exit status is −1 if there are errors parsing the command line.

Exit status is 1 if the agent returns an error.

SEE ALSO

getone(1M), getmany(1M), getnext(1M), snmpd(1M), trap_recv(1M), trap_send(1M), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

setmnt – establish mount table

SYNOPSIS

/etc/setmnt

DESCRIPTION

Setmnt creates the /etc/mnttab table (see mnttab(4)), which is needed for both the mount(1M) and umount commands. Setmnt reads standard input and creates a *mnttab* entry for each line. Input lines have the format:

special filesys

where:

special is the pathname of a special file referring to a device containing a file system (e.g., /dev/dsk/usr).

filesys is the pathname of the directory where the file system is currently mounted. These two strings become the first two strings in the mnttab(4) entry.

To terminate input, type a <CTRL-D>.

FILES

/etc/mnttab

SEE ALSO

mount(1M), mnttab(4).

NOTES

The maximum length of *special* or *filesys* is 32 characters.

NAME

`setuname` - changes machine information

SYNOPSIS

`setuname` [`-s` *name*] [`-n` *node*] [`-t`]

DESCRIPTION

`setuname` changes the parameter value for the system name and node name. Each parameter can be changed using `setuname` and the appropriate option. Only the superuser can use `setuname`.

The options and arguments for this command are:

`-s` Changes the system name. *name* specifies new system name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

`-n` Changes the node name. *node* specifies the new network node name and can consist of alphanumeric characters and the special characters dash, underbar, and dollar sign.

`-t` Temporary change. No attempt will be made to create a permanent change.

Either or both the `-s` and `-n` options must be given when invoking `setuname`.

The system architecture may place requirements on the size of the system and network node name. The command will issue a fatal warning message and an error message if the name entered is incompatible with the system requirements.

NOTES

`setuname` attempts to change the parameter values in two places: the running kernel and, as necessary per implementation, to cross system reboots. A temporary change changes only the running kernel.

NAME

showmount – show all remote mounts

SYNOPSIS

/usr/etc/showmount [-ade] [*host*]

DESCRIPTION

Showmount lists all the clients that have remotely mounted a filesystem from *host*. This information is maintained by the mountd(1M) server on *host*, and is saved across crashes in the file /etc/rmtab. The default value for *host* is the value returned by hostname(1).

OPTIONS

-a Print all remote mounts in the format

hostname : *directory*

where *hostname* is the name of the client, and *directory* is the root of the file system that has been mounted.

-d List directories that have been remotely mounted by clients.

-e Print the list of exported file systems.

FILES

/etc/rmtab

SEE ALSO

hostname(1), mountd(1M), exports(4).

BUGS

If a client crashes, its entry will not be removed from the list until it reboots and executes 'umount -a'.

NAME

shutdown - shut down system, change system state

SYNOPSIS

/etc/shutdown [-y] [-g *grace_period* [-i *init_state*]

DESCRIPTION

This command is executed by the superuser to change the state of the machine. By default, it brings the system to a state where only the operator console has access to the UNIX system. This state is traditionally called "single-user".

The command sends a warning message to all of the terminals (all the people currently logged in) and a final message before it starts actual shutdown activities. By default, the command asks the user at the console for confirmation before it starts shutting down daemons and killing processes. The options are used as follows:

-y pre-answers the confirmation question so the command can be run without user intervention. A default of 60 seconds is allowed between the warning message and the final message. Another 60 seconds is allowed between the final message and the confirmation.

-g*grace_period*
allows the superuser to change the number of seconds from the 60-second default. *grace_period* is expressed in seconds: g300 gives a 5-minute warning.

-i*init_state*
specifies the state that init(1M) is to be put in following the warnings, if any. By default, system state "s" is used.

Possible system states are:

state s, S
Bring the machine to the state traditionally called single-user. The /etc/rcS.d rc scripts are called to do this work. All processes are killed and all file systems other than root are unmounted.

state 1
Bring the machine to the state called the administrator run level. All local file systems will be mounted, and the update daemon will be running. If specified in the /etc/inittab file, optional "administrative" terminals may be enabled.

SEE ALSO

init(1M), inittab(4).

NAME

snmpd – SNMP agent

SYNOPSIS

snmpd [-v][-d] [-p *interval*]

where:

interval is the polling interval in seconds

DESCRIPTION

The SNMP agent is implemented as the snmpd command, which is a daemon process that services requests from an SNMP network management station (NMS). The agent may be included in the list of daemons in the /etc/tcpip.params file to be started and stopped automatically by the rc scripts during changes of system run levels. You also can start or stop the agent through sysadm.

When snmpd is started it disassociates itself from the controlling terminal, reads the configuration files, and begins servicing network requests. The agent listens for requests on the UDP port returned by getservbyname(3N) for "snmp" which defaults to 161.

The agent authenticates the request by verifying that the community string in the request matches one in the /etc/snmpd.communities file and that the level of access granted the community matches the type of request. After authentication, the agent accesses or modifies the requested information in the kernel and then sends a reply to the originator.

When the agent determines it has been restarted, or when an interface changes state, or when a request fails authentication it sends traps to all the hosts specified in /etc/snmpd.trap_communities file.

Options are:

-v Use the -v option to force snmpd to remain attached to the controlling terminal and to print additional information about the packets received and transmitted. Use this option to see the information exchange between the agent and a management station.

-d Use the -d option to force snmpd to remain attached to the controlling terminal and to print diagnostic messages on stderr. This option is similar to the -v option, however, it does not print information about packets received and transmitted. Use this option to see any potential error messages without the verbosity of packet exchanges.

-p *interval*
Use this option to set the polling *interval* (in seconds) that snmpd uses to check for changes in the interfaces state. An *interval* of 0 or an empty /etc/snmpd.trap_communities file will disable polling. The default polling interval is 60 seconds.

FILES

snmpd.config
Use this file to override default values for objects that are machine dependent.

snmpd.communities
Use this file to define the list of community strings, host addresses, and access levels recognized by the agent.

`snmpd.trap_communities`
: Use this file to define the list of communities, host addresses, and port numbers where the agent sends traps.

DIAGNOSTICS

If the `-v` or `-d` options are specified output is sent to stdout and stderr, otherwise, all output is sent to `syslogd`.

SEE ALSO

`getmany`(1M), `getnext`(1M), `getone`(1M), `setany`(1M), `syslogd`(1M), `trap_send`(1M), `trap_recv`(1M), `getservbyname`(3N), `snmpd.config`(4), `snmpd.communities`(4), `snmpd.trap_communities`(4).

NAME

spray - spray packets

SYNOPSIS

/usr/etc/spray [-c *count*] [-d *delay*] [-i] [-l *length*] *host*

DESCRIPTION

Spray sends a one-way stream of packets to *host* using RPC, and reports how many were received, as well as the the transfer rate. The *host* argument can be either a name or an internet address.

OPTIONS

-c *count*

Specify how many packets to send. The default value of *count* is the numbers of packets required to make the total stream size 100000 bytes.

-d *delay*

Specify how may microseconds to pause between sending each packet. The default is 0.

-i Use ICMP echo packets rather than RPC. Since ICMP automatically echos, this creates a two way stream. This requires a raw socket, so you must be "root" to do this.

-l *length*

The *length* parameter is the numbers of bytes in the Ethernet packet that holds the RPC call message. Since the data is encoded using XDR, and XDR only deals with 32 bit quantities, not all values of *length* are possible, and spray rounds up to the nearest possible value. When *length* is greater than 1514, then the RPC call can no longer be encapsulated in one Ethernet packet, so the *length* field no longer has a simple correspondence to Ethernet packet size. The default and minimum value of *length* is 86 bytes (the size of the RPC and UDP headers)

SEE ALSO

ping(1C), sprayd(1M), icmp(6P).

NAME

sprayd – spray server

SYNOPSIS

/usr/etc/rpc.sprayd

DESCRIPTION

Rpc.sprayd is a server (daemon) that records the packets sent by spray(1M). The rpc.sprayd server is normally invoked by inetd(1M).

SEE ALSO

inetd(1M), spray(1M).

NAME

statd – network status monitor

SYNOPSIS

/usr/etc/rpc.statd

DESCRIPTION

Statd is an intermediate version of the status monitor. It interacts with lockd(1M) to provide the crash and recovery functions for the locking services on NFS.

A server may have more than one network interface name. When configuring a client, use the server's primary name. Otherwise the client will fail to recover its locks when the server reboots. To determine the primary network interface name, execute hostname(1M) on the server.

FILES

/etc/sm
/etc/sm.bak
/etc/state

SEE ALSO

lockd(1M), statd(4).

BUGS

The crash of a site is only detected upon its recovery.

NAME

`strace` – print STREAMS trace messages

SYNOPSIS

`strace` [*mid sid level*] ...

DESCRIPTION

`Strace` without arguments writes all STREAMS event trace messages from all drivers and modules to its standard output. These messages are obtained from the STREAMS log driver (`log`(7)). If arguments are provided, they must be in triplets of the form *mid, sid, level*, where *mid* is a STREAMS module id number, *sid* is a sub-id number, and *level* is a tracing priority level. Each triplet indicates that tracing messages are to be received from the given module/driver, sub-id (usually indicating minor device), and priority level equal to or less than the given level. The token `all` may be used for any member to indicate no restriction for that attribute.

The format of each trace message output is:

seq time ticks level flags mid sid text

seq	trace sequence number
time	time of message in hh:mm:ss
ticks	time of message in machine ticks since boot
level	tracing priority level
flags	E : message is also in the error log F : indicates a fatal error N : mail was sent to the system administrator
mid	module id number of source
sid	sub-id number of source
text	formatted text of the trace message

Once initiated, `strace` will continue to execute until terminated by the user.

EXAMPLES

Output all trace messages from the module or driver whose module id is 41:

```
strace  41 all all
```

Output those trace messages from driver/module id 41 with sub-ids 0, 1, or 2:

```
strace  41 0 1  41 1 1  41 2 0
```

Messages from sub-ids 0 and 1 must have a tracing level less than or equal to 1. Those from sub-id 2 must have a tracing level of 0.

SEE ALSO

`strerr`(1M), `log`(7).
STREAMS Programmer's Guide for the DG/UX System.

NOTES

Due to performance considerations, only one `strace` process is permitted to open the STREAMS log driver at a time. The log driver has a list of the triplets specified in the command invocation, and compares each potential trace message against this list to decide if it should be formatted and sent up to the `strace` process. Hence, long lists of triplets will have a greater impact on overall STREAMS performance. Running `strace` will have the most impact on the timing of the modules and drivers generating the trace messages that are sent to the `strace` process. If trace messages are generated faster than the `strace` process can handle them, some of the messages

will be lost. This last case can be determined by examining the sequence numbers on the trace messages output.

NAME

strclean - STREAMS error logger cleanup program

SYNOPSIS

strclean [-d *logdir*] [-a *age*]

DESCRIPTION

Strclean is used to clean up the STREAMS error logger directory on a regular basis (for example, by using cron(1M)). By default, all files with names matching error.* in /usr/adm/streams that have not been modified in the last three days are removed. A directory other than /usr/adm/streams can be specified with the -d option. The maximum age in days for a log file can be changed using the -a option.

EXAMPLES

```
strclean -d /usr/adm/streams -a 3
```

has the same result as running strclean with no arguments.

FILES

/usr/adm/streams/error.*

SEE ALSO

cron(1M), strerr(1M).
STREAMS Programmer's Guide for the DG/UX System.

NOTES

strclean is typically run from cron(1M) on a daily or weekly basis.

NAME

strerr – STREAMS error logger server

SYNOPSIS

strerr

DESCRIPTION

Strerr receives error log messages from the STREAMS log driver (log(7)) and appends them to a log file. The error log files produced reside in the directory /usr/adm/streams, and are named error.*mm-dd*, where *mm* is the month and *dd* is the day of the messages contained in each log file.

The format of an error log message is:

seq time ticks flags mid sid text

where:

seq	error sequence number
time	time of message in hh:mm:ss
ticks	time of message in machine ticks since boot
flags	T : the message was also sent to a tracing process F : indicates a fatal error N : send mail to the system administrator
mid	module id number of source
sid	sub-id number of source
text	formatted text of the error message

Messages that appear in the error log are intended to report exceptional conditions that require the attention of the system administrator. Those messages which indicate the total failure of a STREAMS driver or module should have the F flag set. Those messages requiring the immediate attention of the administrator will have the N flag set, which causes the error logger to send the message to the system administrator via mail (1).

Once initiated, strerr will continue to execute until terminated by the user. Commonly, strerr would be executed asynchronously.

FILES

/usr/adm/streams/error.*mm-dd*

SEE ALSO

strace(1M), log(7).
STREAMS Programmer's Guide for the DG/UX System.

NOTES

Only one strerr process at a time is permitted to open the STREAMS log driver.

If a module or driver is generating a large number of error messages, running the error logger will cause a degradation in STREAMS performance. If a large burst of messages occurs in a short time, the log driver may not be able to deliver some of the messages. This situation is indicated by gaps in the sequence numbering of the messages in the log files.

NAME

sttydefs - maintain line and hunt settings for TTY ports

SYNOPSIS

/usr/sbin/sttydefs -a *ttylabel* [-b] [-n *nextlabel*]
[-i *initial-flags*] [-f *final-flags*]

/usr/sbin/sttydefs -l [*ttylabel*]

/usr/sbin/sttydefs -r *ttylabel*

where:

ttylabel	is the name of a record in /etc/ttydefs
nextlabel	is the next record of a hunt sequence
initial-flags	is the list of stty(1) flags used for login
final-flags	is the list of stty(1) flags used after login

DESCRIPTION

sttydefs is an administrative command that maintains the line settings and hunt sequences for the system's TTY ports, by making entries in and deleting entries from the /etc/ttydefs file. It is used as an adjunct to the Service Access Facility (see sac(1M) and ttymon(1M)).

sttydefs with a -a or -r option may be successfully invoked only by a privileged user. sttydefs with the -l option may be invoked by any user on the system.

Options are:

-l Display on the standard output the record from /etc/ttydefs whose TTY label matches the specified *ttylabel*. If no *ttylabel* is specified, display the entire contents of /etc/ttydefs. sttydefs will verify that each entry it displays is correct and that each entry's *nextlabel* field references an existing *ttylabel*. Any errors found during the verification process will produce self-explanatory messages on the standard output.

-a *ttylabel*
Add a record to the /etc/ttydefs file, using *ttylabel* as its label.

-r *ttylabel*
Remove any record in the /etc/ttydefs file that has *ttylabel* as its label.

-b Specify that the "autobaud" feature should be enabled. Autobaud allows the system to set the line speed of a given TTY port by the line speed of the device connected to the port without the user's intervention.

-n *nextlabel*
Specify the value to be used in the *nextlabel* field in /etc/ttydefs. This value identifies the next record in a "hunt sequence" to be tried if a user indicates that the line speed is wrong (by pressing the BREAK key) while logging in. If this option is not specified, sttydefs will set *nextlabel* equal to *ttylabel*.

-i *initial-flags*
Specify the value to be used in the *initial-flags* field in /etc/ttydefs. *initial-flags* must be in a format recognized by the stty(1) command. These termio(7) flags are used by ttymon(1M) when searching for the correct baud rate. They are set prior to writing the prompt. If this option is not specified, sttydefs will set *initial-flags* equal to the termio(7) flag 9600.

-f *final-flags*
Specify the value to be used in the *final-flags* field in /etc/ttydefs. *final-*

flags must be in a format recognized by the `stty`(1) command. *final-flags* is the list of `termio`(7) settings used by `ttymon`(1M) after receiving a successful connection request and immediately before invoking the service on the port. If this option is not specified, `sttydefs` will set *final-flags* equal to the `termio`(7) flags `9600` and `sane`.

The `-l`, `-a`, and `-r` options are mutually exclusive. The `-b`, `-n`, `-i`, and `-f` options can be used only in conjunction with the `-a` option.

EXAMPLES

```
sttydefs -l
```

List all the entries in the `/etc/ttydefs` file and print an error message for each invalid entry that is detected.

```
sttydefs -l 9600
```

Request information for a single label in the `/etc/ttydefs` file. The output of this command would look like the following:

```
--------------------------------------------------
9600:9600 hupcl:9600 sane ixany tab3 erase ^h::4800
--------------------------------------------------

ttylabel:       9600
initial flags:  9600 hupcl
final flags:    9600 sane ixany tab3 erase ^h
autobaud:       no
nextlabel:      4800
```

```
sttydefs -a 1200 -n 2400 -i 1200 -f "1200 sane"
sttydefs -a 2400 -n 4800 -i 2400 -f "2400 sane"
sttydefs -a 4800 -n 9600 -i 4800 -f "4800 sane"
sttydefs -a 9600 -n 1200 -i 9600 -f "9600 sane"
```

Add the labels `1200`, `2400`, `4800`, and `9600`, putting them in a circular hunt list.

FILES

`/etc/ttydefs` TTY settings file
`/etc/.ttydefs`
temporary file

DIAGNOSTICS

Exit Codes

If successful, `sttydefs` will exit with a status of `0`. If an error occurs during its operation, `sttydefs` will print an error message to standard error and exit with a status of `1`. An error in the command line will cause `sttydefs` to print a usage message to standard error and exit with a status of `2`.

Error Messages

`User not privileged for operation.`

An attempt was made by someone other than the super-user to add (`-a`) or remove (`-r`) an entry from `/etc/ttydefs`.

`Version number is incorrect or missing.`

The `/etc/ttydefs` file is corrupt, does not contain a "VERSION=" line, or specifies an unrecognized version ID.

`Tempfile busy; try again later.`

Someone else is currently using `sttydefs` to update `/etc/ttydefs`. Otherwise, an invocation of `sttydefs` crashed, or the system crashed while `sttydefs` was running, leaving behind an extraneous `/etc/.ttydefs` temporary file.

`Ttylabel` *ttylabel* `not found`

An attempt to list (`-l`) the record *ttylabel* failed because that entry does not exist in `/etc/ttydefs`.

`Ttylabel` *ttylabel* `already exists.`

An attempt to add (`-a`) the record *ttylabel* failed because that entry already exists in `/etc/ttydefs`.

`Ttylabel` *ttylabel* `does not exist.`

An attempt to remove (`-r`) the record *ttylabel* failed because that entry does not exist in `/etc/ttydefs`.

Other error messages should be self-explanatory, reporting either usage errors or failure of a system call or library function.

SEE ALSO

`sac`(1M), `stty`(1), `ttymon`(1M), `termio`(7).

System Administrator's Guide, "Terminal Line Settings."

NAME

swapon - specify additional devices for system paging

SYNOPSIS

```
/etc/swapon -a
/etc/swapon name ...
```

DESCRIPTION

Swapon specifies additional devices for paging. The system begins by paging on just one device so that only one disk is required at bootstrap time. Calls to swapon normally occur in the system rc script chk.system in the /usr/sbin/init.d directory. By default, this will invoke swapon the first time you change init run levels. See the file /etc/dgux.params to set the argument to the swapon command in the rc script chk.setup.

Normally, the -a argument is given. This makes available all devices marked as swap devices in /etc/fstab. A secondary swap device entry in etc/fstab would be as follows:

```
· /dev/dsk/swap1   swap_area   swap sw x 0
```

The second form of swapon makes the individual block-special devices specified on the command line available to the system for swap allocation.

FILES

/dev/dsk/* Potential paging devices

SEE ALSO

swapon(2), fstab(4).

BUGS

You cannot stop paging on a device. Therefore, you cannot use swapon for devices that may be dismounted during system operation.

NAME

syac_routes - Change SYAC routing information

SYNOPSIS

syac_routes [-a] [-d *dir*] [-f *file*] *device*

where:

device The full pathname of a SYAC device.

DESCRIPTION

The syac_routes command changes the routing information used by the specified SYAC, which must be a VTC. The old routing information is flushed from the board unless the -a option is specified. The new routing information is taken from the file specified in the /etc/tcload/vtc.addrs configuration file for the SYAC device unless either the -f or the -d option is specified.

Options are:

-a Add the new routing information to the information currently in use. By default, the current routing information is flushed before the new information is communicated to the board.

-f *file* Read the new routing information from the specified file instead of from the file specified in /etc/tcload/vtc.addrs. The file format should be identical to that of /etc/gateways (see gateways(4)). If default is specified for the file name, then the routing information currently in use by the host computer will be communicated to the board.

-d *dir* Look in the specified directory for the vtc.addrs file instead of /etc/tcload. If both the -d option and the -f option are specified, then the -f option takes precedence.

EXAMPLES

```
syac_routes -a /dev/async/syac@60(60000000)
```

This will read the routing file specified in /etc/tcload/vtc.addrs and communicate the routing information found there to the board. The current routing information will not be flushed.

```
syac_routes -f default /dev/async/syac@60(60000000)
syac_routes -f /etc/test_file /dev/async/syac@60(60000000)
```

The first example will communicate the routing information in use by the host computer to the specified SYAC. The second example will read the file /etc/test_file and communicate the routing information found there to the board. Both examples will flush the current routing information before communicating the new routing information to the board.

```
syac_routes -d /tmp/test /dev/async/syac@60(60000000)
```

This will read the routing information specified in /tmp/test/vtc.addrs and will communicate the routing information found there to the board.

FILES

/etc/tcload/vtc.addrs SYAC VTC Configuration file

DIAGNOSTICS

Exit status is 0 for success, non-zero otherwise.

SEE ALSO

syac_ttyaddrs(1M), gateways(4), vtc.addrs(4M), syac(7).

NAME

syac_ttyaddrs - set tty specific internet addresses

SYNOPSIS

syac_ttyaddrs [-b on|off] [-p on|off] *device ttyname ... inet_addr*

where:

device The full pathname of the syac device which owns the specified tty(s).

ttyname The full pathnames of the tty(s) which should respond to the specified internet address.

inet_addr The internet address for the tty(s), expressed in dot notation (see inet(3N)).

DESCRIPTION

The syac_ttyaddrs command operates on tty(s) controlled by a SYAC VTC device. Normally, all ttys associated with a VTC respond to telnet connections to the VTC default internet address (see vtc.addrs(4M)). The syac_ttyaddrs command directs the named tty(s) to respond to telnet connections to the internet address specified on the command line. The behavior of all the other ttys controlled by the VTC is unaffected.

The ability to associate specific internet addresses with specific ttys is useful when using passive devices, such as printers, with the VTC. In order to access a device, a program must know the entry in /dev that is associated with the device. For the case of a device using a permanent telnet connection to a VTC (as via a termserver), the entry in /dev would be a tty entry, such as /dev/tty56. The system must ensure that the telnet connection for such a device is always associated with a specific tty associated with a VTC. The only way to do this is to assign that particular tty a specific internet address and have the device connect to this particular internet address via the telnet protocol. In this way the device will always be associated with the proper tty.

Tty specific internet addresses can be assigned at any time with the syac_ttyaddrs command and are active until the system is rebooted or the SYAC is reset. The system will also assign tty specific internet addresses during system boot based on the contents of /etc/tcload/vtc.addrs (see vtc.addrs(4M)).

The SYAC board performs some aspects of input processing for the host computer. By default, the input processing performed by the SYAC (and the input processing performed by the host) is not affected by the state of telnet binary mode, and can only be enabled and disabled by changing the current line discipline settings (see termio(7)). Specific tty lines can be configured so that when the telnet connection is negotiated into telnet binary mode, the input processing performed by the SYAC is disabled. Input processing performed by the host is unaffected. The vast majority of applications will not require this behavior, however, this option is supported for applications which may require it.

Options are:

-b on|off
: If on, the VTC will attempt to negotiate telnet binary mode when a connection is established for any of the tty lines specified on the command line. If off, the VTC will not attempt to negotiate telnet binary mode. By default, the VTC will attempt to negotiate telnet binary mode for all lines when a connection is established.

-p on|off
: If on, onboard input processing will not be affected by the state of telnet binary mode. If off, onboard input processing will be disabled whenever

telnet binary mode is negotiated on. By default, onboard input processing is not affected by the state of telnet binary mode.

EXAMPLES

syac_ttyaddrs /dev/async/syac@60(60000000) /dev/tty34 128.222.3.112

This sets the internet address for /dev/tty34 to 128.222.3.112. An error is generated if the specified SYAC does not refer to a VTC device or if the specified tty(s) are not controlled by the specified SYAC.

FILES

`/etc/tcload/vtc.addrs` SYAC VTC Configuration file

DIAGNOSTICS

Exit status is 0 for success, non-zero otherwise.

SEE ALSO

`syac_routes`(1M), `inet`(3N), `vtc.addrs`(4M), `syac`(7), `termio`(7).

NAME

syacdb – syac debugger utility program

SYNOPSIS

syacdb [*device*] ...

where:

device Name of systech controller to use

DESCRIPTION

The syacdb command is a set of utilities than can be used to provide information about the hardware, software,and configuration of the Systech controllers. If no device is specified then the program will attempt to use the default syac device which is /dev/async/syac@60(60000000). An alternname device may be specified on the command line.

EXAMPLES

syacdb /dev/async/syac@62(60040000)

COMMANDS

!

The **!** command can be used to issue commands to the parent shell.

?

The **?** command is the same as the help command. This command will print out a list of valid commands.

cch [channel-id]

The **cch** command is used to look at the cluster channel data structure. Two pages of output are produced information of the receive structure, the transmit structure and general information. If a channel id is specicified then the system attempts to gain information about that channel. Otherwise the program uses the current channel id.

cdump [channel_id] [starting_addr]

The **cdump** command is used to dump memory from a cluster controller. The channel id and starting addresses are mandatory parameters the first time the command is used. Once the user has specified a channel id, the program will continue to use the same channel until a new chanel is specified. The command will dump out channel memory in 128 byte blocks. If the address is not given then the program will use the current memory location +1 as the starting address. Sample output is shown below.

```
100080: 00 10 03 c0 00 20 04 04 00 10 04 0a 00 00 0f da ..... .....
100090: 00 00 0f da 00 00 15 b0 00 10 43 90 00 10 66 98 .....C...f.
1000a0: 00 00 0f da 00 00 0f da 00 00 0f da 00 00 0f da ..... .....
1000b0: 00 00 0f da 00 00 0f da 00 00 0f da 00 00 0f da ..... .....
1000c0: 00 00 0f da 00 00 0f da 00 00 0f da 00 00 0f da ..... .....
1000d0: 00 00 0f da 00 00 0f da 00 00 0f da 00 00 0f da ..... .....
1000e0: 00 00 0f da 00 00 0f da 00 00 0f da 00 00 0f da ..... .....
```

clog [cluster_id] [device]

The **clog** command is used to get general logging information about the cluster controller. The cluster id is a mandatory parameter the first time the command

is invoked. From then on the program will use the id of the last invocation of the program. If not specified, the program uses the device specified when the program was invoked. The following is an example of a call to clog:

Model	HPS-7082-030
Firmware	90-070371-7-01A
Running	90-070434-7-01C+SEQ
Device type	69
Station ID	01
Real-time clock ticks	120049
Idle task counter	120049
Out of memory errors	0
Miscellaneous errors	0
Network packets send	2208
Network packets received	2224
No response errors	0
Bad response errors	0
Offline errors	0
Network reconfigurations	1
ROM size	65536
ROM checksum	0x0714
End of RAM	1179648
Failed ports	none

`dloop` [cdlnrstq] [device] [limit] [bcount] [bsize] [time]
The **dloop** command performs DMA loopback testing between the host computer and the Systech host adapter. A block of data is sent to the board, read back into the host and verified.

`options`
One or more of the options must be selected.

`c` This option will disable the verification part of the test. The blocks will be transmitted and received but no check will be made that the data is the same.

`d` This option allows the user to override the default device which is /dev/async/syac@60(60000000).

`l` This option allows the user to set a limit on the characters per second sent by the test. If this option is used the user must specify a limit on the command line.

`n` This option allows a maximum block count to be set. If this option is used the user must specify a block count on the command line. The default is to send an unlimited number of blocks with the test bounded by time.

`r` This option enables sending random size blocks from 1 to 512 bytes in length. The default is to send fixed size blocks.

`s` This option specifies that fixed block lengths are to be used. The maximum block size allowed is 512 bytes. If this option is used then the user must specify a block size on the command line.

`t` This option sets the running time for the test. The default time is 30 seconds. If this option is used then the user must specify the time in seconds on the command line.

`q` This option tells the program not to send results to standard output

unless there is an error.

If two or more conflicting options are specified then the first option on the command line takes effect.

Examples

Run the loopback test for 60 seconds using random size blocks.

dloop rt 60

Run the test sending 200 blocks that are 256 bytes long.

dloop ns 200 256

dump [starting_addr] [dump_size]
The **dump** command allows the user to display portions of the syac memory. If no starting address is specified then the dump will be started at either the start of syac memory if memory had not been dumped previously or the next location after the last location dumped. The default dump size is 128 bytes. The maximum size of memory to be dumped at one time is 512 bytes. The format of the dump is the same as for the cdump command.

exit
The **exit** command terminates the program.

help
The **help** command prints a list of valid commands for the **syacdb** program

log
The **log** command command is used to get general logging information about the host adapter. Sample output from the **log** command is shown below.

Model HPS-6945
Firmware 90-070154-6-05B Board assy XX-XXXXXX-X-XXX
Device type 4

Station ID	ff	Option switches	0xfc
Real-time clock ticks	370819	Idle task counter	411888196
Bytes DMAed to HPS	128822	Bytes DMAed from HPS	1097777
DMA XACK timeouts	0	IOCB processed	2872
DMA bus req. timeouts	0	Response queue full	0
No memory errors	0	Response queue full	0
Network queue full	0	Miscellaneous errors	0
Packets sent	6502	Packets received	6410
No response errors	0	Bad response errors	0
Offline errors	0	Reconfigurations	8
reserved	0	ROM size	32768
ROM checksum	0x6d87	End of RAM	0x180000
Real time clock (HZ)	20	reserved	0

Set ram start to 0x100000 and ram_end to 0x17ffff

netc
The **netc** program prints out the configuration of syac controllers and cluster

controllers that exist on the network. For each controller the program will print out the device type, network address along with the state of lines if appropriate. The following gives sample output from the **netc** program.

NET ADDR	STATE	DEV TYPE	FW RE	HOST ID
1	Normal Online	HPS 7082-030	01A	n/a
ff	Normal Online	HPS 6945	05B	0

`partitions`
The **partitions** command prints a table showing the breakup of on-board memory into certain sizes pieces. The program will show the number of the pieces, the size and current location of the memory chunks. An sample output from the partitions command is shown below.

Part	Size	Name	Max	# Curr Entries	Start	End
1	512	MEM_512	506	328	0x00140b94	0x0017ff94
3	32	MEM_32	183	172	0x0012f834	0x00130f14
4	16	MEM_16	60	10	0x0012f474	0x0012f834
2	32	MEM_32	183	172	0x0012f834	0x00130f14
			16	16	0x001005ec	0x001009ec

`queues`
The **queues** command provides information about the state of the on-board queues. Information such as number of entries and number of tasks waiting are provided. An example of output from the queues command is shown command.

Queue	# Cur Entries	# Tasks Waiting	Name	Max Entries
244	0	1	DIAG NET Q	16
243	0	1	DIAG IOCB Q	16
54	175	-	MEM 512 Q	175
41	0	1	ASYNC IOCB Q	26

`rloop a[cdlnrstq] addr [device][limit][bcount][bsize][time]`
The **rloop** command performs DMA loopback testing between the host computer and the remote cluster controller. A block of data is sent to the controller, read back into the host and verified.

`options`
One or more of the options must be selected.

`a` This option is used to tell the program which cluster controller to run the loopback test through.

`c` This option will disable the verification part of the test. The blocks will be transmitted and received but no check will be made that the data is the same.

`d` This option allows the user to override the default device which is /dev/async/syac@60(60000000).

l This option allows the user to set a limit on the characters per second sent by the test. If this option is used the user must specify a limit on the command line.
n This option allows a maximum block count to be set. If this option is used the user must specify a block count on the command line. The default is to send an unlimited number of blocks with the test bounded by time.
r This option enables the sending of random size blocks from 1 to 512 bytes in length. The default is to send fixed size blocks.
s This option specifies that fixed block lengths are to be used. The maximum block size allowed is 512 bytes. If this option is used then the user must specify a block size on the command line.
t This option sets the running time for the test. The default time is 30 seconds. If this option is used then the user must specify the time in seconds on the command line.
q This option tells the program not to send results to standard output unless there is an error.

If two or more conflicting options are specified then the first option on the command line takes effect.

Examples

Run the remote loopback test on cluster controller 2 for 60 seconds using random size blocks.

```
rloop art 2  60
```

Run the remote test on cluster controller 1 sending 200 blocks that are 256 bytes long.

```
rloop ans 1 200 256
```

search [pattern] [starting_addr] [ending_addr]
The **search** command is used to search through host adapter memory to locate a pattern. The search pattern is initialized to 00. The initial starting and ending addresses to search are set to the top and bottom of memory. The starting address is set to the next location after a successful search. If a search was not successful then the starting address is not modified. A pattern may be between 1 and 4 bytes in length and leading zeroes must be specified.

setcch [addr]
The **setcch** command is used to get or set the cluster channel address.

setl [length]
The **setl** command is used to get or set the channel length field.

settcb
The **settcb** command is used to get or set the task control block address.

show
The **show** command is used to print out internal variables used by the progrm.

Information like the starting search address and the board type will be printed out. A number of the variables are initialized to zero and not set until the appropriate command has been called first to set the variable. For example prior to the first invocation of the queue command the queue addr field is set to 0x0. A sample output from the **show** command is given below.

```
Global variables:
  hps_intel      = 0x0
  board_type     = 0x4
  os_type        = 0x0
  ram_start      = 0x100000
  ran_end        = 0x17ffff
  tcb_addr       = 0x127dae
  tcb_len        = 0x50
  config_tasks   = 0x11
  ch_addr        = 0x0
  ch_len         = 0x390
  cch_addr       = 0x0
  queue_addr     = 0x0
  part_addr      = 0x0
Other current variables:
  remaddr        = 0x0
  station        = 0x800
  dumpaddr       = 0x0;
  dumplen        = 0x0
  search data    = 0x0
  search size    = 0x0
  search chart   = 0x0
  search start   = 0x0
  search end     = 0x0
```

`hpsstat` [dcpfioexsta] [device] channel_number
The **hpsstat** command is used to retrieve channel parameters for an open channel from the host adapter. The default host adapter is /dev/async/syac@60(60000000)).

`options`
One or more options must be selected:

`t` This option must be used with either the i or o option. This will cause the appropriate translate table to be printed.
`c` This option is used to display system configuration. This option conserns the entire host adapter and not just an open channel.
`p` This option causes the current baud rate, number of bits, parity and stop bits to be printed out.
`f` This option causes the display flow control parameters to be printed.
`i` This will cause the input parameters to be printed.
`o` This will cause the output parameters to be printed.
`e` This will cause echo parameters to be printed.
`x` This option is to display extra parameters which have not been implemented in the current release of the Systech firmware.
`s` This will print out the display channel status.
`d` This allows the user to set the device name and override the default

device.
a This allows all options to be set.
Sample output from the stat program is listed below.

ASYNCH PROTOCOL

	Baud	Char	Len	Parity
Receive:	300	8	none	1 stop bit
Transmit:	300	8	none	1 stop bit

`tasks [ task-id ]`
The **tasks** command is used to get information about an individual task or about all of the tasks on the host adapter. If the task id is specified the information includes the contents of the registers. Sample output from the command is given below.

Task	TCB	Pri	State
4 (loader task)	(00127e4e)	11	RDY
12 (netman)	(00127e9e)	14	SUS QUE: 10. (NETW Q)
5 (hst_Rsp_task)	(00127eee)	100	SUS QUE: 5. (RESP Q)
255 (idle task)	(001282ae)	255	RDY

`usage`
The **usage** command is used to measure the load on the host adapter. The program will report the percentage of time the board is idle.

`ver`
The **ver** program is used to get the version of the terminal control software (TCS) that is running on the host adapter. When this program is invoked, a small menu appears which allows the user to get the version information, change the host adapter to allow getting version information on a different adapter and to quit the program. The user makes a choice by entering the menu number and pressing the return key. Sample version information and the user menu are shown below.

==TCS VERSION INFORMATION==
Title: HPS-TCS-VME
Version: 04H
Part Number: 90-070054-5
Creation Date: Mon Oct 8 09:22:44 PDT 1990
Description: HPS-6945 TCS and Diag Apps, Net Man

Which version information do you want to display?
0 - Quit
1 - TCS
2 - Change host adapter (currently /dev/async/syac@60(60000000))
Your choice (enter a number)

FILES

`/usr/sbin/syacdb` Executable file for utility
`/usr/lib/tcload/models/syac/dloop` Executable file for the dloop command
`/usr/lib/tcload/models/syac/rloop` Executable file for the rloop command

/usr/lib/tcload/models/syac/hpsver Executable file for the ver command
/usr/lib/tcload/models/syac/stat Executable file for the stat command

SEE ALSO

syac(7).

NOTES

Certain commands such as clog cannot be run until the netc command has been run.

The netc command has no meaning on a VAC16 and will cause the program to exit.

NAME

syacdump – dump syac memory to a file

SYNOPSIS

syacdump *pathname*

where:

pathname

The pathname of a syac control node

DESCRIPTION

The syacdump command dumps the syac onboard memory to the file hpscore for later analysis.

EXAMPLES

```
syacdump syac@60(60000000)
```

FILES

/dev/async/syac* syac control nodes
hpscore The memory dump file

DIAGNOSTICS

Exit status is 0 if the syac is dumped, 1 if not.

SEE ALSO

syac(7), syacdb(1M).

NOTE

syacdump cannot dump VTCs.

NAME

sync – update the super-block

SYNOPSIS

sync

DESCRIPTION

Sync executes the sync system primitive. You can call sync to ensure all disk writes have been completed before the processor is halted.

See sync(2) for details on the system primitive.

SEE ALSO

halt(1M), update(1M), sync(2).

NAME

sysadm, asysadm, xsysadm – menu-driven system administration interface

SYNOPSIS

sysadm [-l] [*menu-alias*]

asysadm [-l] [-m *menu*] [-o *operation*] [*menu-alias*]

xsysadm [*X11-options*]

DESCRIPTION

The asysadm and xsysadm commands provide menu-driven interfaces to system administration functions. sysadm is a generic invocation name that uses the absence or presence of the WINDOWID environment variable to determine whether to start asysadm or xsysadm. References to sysadm are applicable to both asysadm and xsysadm.

asysadm is designed for use on a character-based terminal or terminal emulator. This version of the command presents you with menus and scrolling interactive queries to help you choose and execute the commands to administer the system.

xsysadm uses the X11 window system on a graphics workstation. When using this version of the command, you may use the mouse to select functions to be performed from menus, and dialog windows appear in which information is accepted.

Both interfaces are designed to be consistent while taking advantage of the capabilities of the display device. In both cases you select an object or class of objects to be managed, possibly refine your selection through additional menus until you select a specific object type, and then you select an operation to be performed. Once you make this selection, sysadm prompts you to enter whatever information is necessary to carry out the operation. When all information has been obtained, the information is verified, you are asked to confirm that you want the operation carried out, and then the operation is performed.

Most operations are carried out by invoking

adm*object* -o *operation options*

for the given object class and operation. See adm*object*(1M) for information on management of a specific object.

The method for traversing the menu hierarchy and interacting with the menu system is described in idi(1). For a complete explanation of the sysadm program, see *Managing the DG/UX System*.

Options

Options to sysadm depend on which command you are using.

xsysadm uses only those options which are defined by the X11 window system. See X(1).

asysadm accepts the following options:

-l List the available *menu-alias*es.

-m *menu* Select *menu* as the first menu to be shown, bypassing the Main Menu. The form of a *menu* is described in idi(1).

-o *operation*
Perform *operation* on *menu*.

asysadm accepts a *menu-alias*, which is sought in one of the system-wide or personal alias files and expanded to -m and optionally -o options. The alias directs you

straight into a lower-level menu, without seeing the Main Menu.

If you invoke `asysadm` with neither the `-m` option nor a *menu-alias*, the top level menu, the Main Menu, of system administration objects is displayed.

Resources

X11 resources for `xsysadm` use the class name "Idi" and the default instance name "sysadm". See `idi`(1) for a list of resource names.

Configuration

The `sysadm` program offers several levels of customization. Administrators may add site-specific menus and operations by creating `/usr/lib/sysadm/`*locale*`/sysadm.whatever.rc` file(s), and host-specific menus and operations by creating `/etc/sysadm/`*locale*`/sysadm.whatever.rc` file(s) containing the new items. Any such file(s) will be read along with the default `/usr/lib/sysadm/`*locale*`/sysadm.rc` to create the `sysadm` menu tree. See `idl`(4) for a description of the format of this file.

An individual user may add user-specific menus and operations by creating a `.sysadmrc` file in the home directory or the current directory.

Additionally, host-specific or user-specific menu aliases may be added via the `/etc/sysadm/`*locale*`/sysadm.alias` or `$HOME/sysadm.alias` files.

Users may customize `xsysadm` by modifying the values of the resources used by `idi`. See `idi`(1) for a list of valid resources and `X`(1) for a discussion of how to modify the resources.

EXAMPLES

```
# asysadm adduser
```

This command line takes you directly to the `add` operation for objects of type `user`, because the line

```
adduser     -m :user:login:add
```

appears in one of the system-wide alias files. You will begin interacting with the dialogue to add a new user account.

In order to change `xsysadm`'s font size, add the following lines to the appropriate X resource file (see `Xdefaults`(5)):

```
sysadm*fontList:                    *helvetica-medium-r*18*
sysadm*logText.fontList:            *courier-medium-r*18*
sysadm*reportText.fontList:         *courier-medium-r*18*
sysadm*selectionBox.labelFontList:  *helvetica-medium-r*18*
sysadm*selectionBox.textFontList:   *courier-medium-r*18*
sysadm*XmList.fontList:             *courier-medium-r*18*
```

These lines specify that `xsysadm` should use 18 point fonts, instead of the default 14 point fonts.

If you often manage more than one system from the same graphics display, you may wish to change `xsysadm`'s title strings to indicate which system is being managed. To do so, you could add the following line to `$HOME/.sysadmrc`:

```
set TitleSuffix = " (`hostname`)"
```

This specifies that the name of the current host be added to the end of all title strings in `xsysadm`.

FILES

/usr/lib/sysadm/*locale*/sysadm*.rc
Top-level file for host-independent DG/UX description files.

/etc/sysadm/*locale*/sysadm*.rc
Top-level file for host-dependent DG/UX description files.

/usr/opt/*package*/lib/sysadm/*locale*/sysadm.rc
Top-level file for package-specific host-independent description files.

/opt/*package*/etc/sysadm/*locale*/sysadm.rc
Top-level file for package-specific host-dependent description files.

$HOME/.sysadmrc
User-dependent description file.

./.sysadmrc
Invocation-dependent description file.

/usr/lib/sysadm/*locale*/sysadm.alias
Host-independent system-wide alias file.

/etc/sysadm/*locale*/sysadm.alias
Host-dependent system-wide alias file.

$HOME/sysadm.alias
Personal alias file.

/var/adm/log/sysadm.log
Log file.

/usr/opt/X11/lib/app-defaults/Idi
X11 default resources file.

ENVIRONMENT

LANG Used to determine the *locale* (the default locale is C).

HOME Used to locate files that are expected in the user's home directory.

PATH Used to locate standard system commands, adm* commands, and certain utility programs.

SEE ALSO

diskman(1M), adm*object*(1M), idi(1), idl(4), X(1), Xdefaults(5), *Installing the DG/UX System*, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NOTES

The sysadm command invokes idi to perform menu and query traversal. Therefore, in order to kill a sysadm session via kill(1) or dg_kill(1), you must kill the idi process.

In DG/UX 5.4, sysadm replaces the previous command by that name. The earlier version is temporarily available under the name osysadm. The earlier version will not be available in future releases.

The new sysadm provides all the functions of osysadm but also provides

* A more object-oriented approach to system management. You select an object to be managed, then select the operation to perform on the object.
* Multiple user interfaces, including an ASCII line-oriented interface and and OSF/Motif-based interface.
* Customizable access permissions on menus and operations. This allows ordinary users access to some system management functions.

NAME

sysdef – output system definition

SYNOPSIS

/etc/sysdef [*system_namelist*]

DESCRIPTION

Sysdef outputs a duplicate of the system file used to build the system specified in *system_namelist*. It lists all hardware devices as well as pseudo devices, network protocols, stream modules and the values of all tunable parameters. It generates the output by analyzing the named operating system file (*system_namelist*) and extracting the configuration information from the name list itself.

FILES

/dgux default operating system image (where the system namelist is)

SEE ALSO

config(1M), master(4), system(4).
Installing the DG/UX System, *Customizing the DG/UX System*, *Managing the DG/UX System*.

NOTES

Sysdef no longer depends on master.d, which is a directory of master files.

NAME

`syslogd` – log systems messages

SYNOPSIS

`/etc/syslogd [ -f`*configfile* `] [ -m`*markinterval* `] [ -d ]`

DESCRIPTION

`Syslogd` reads and logs messages into a set of files described by the configuration file `/etc/syslog.conf`. Each message is one line. A message can contain a priority code, marked by a number in angle braces at the beginning of the line. Priorities are defined in `<sys/syslog.h>`. `Syslogd` reads from the UNIX domain socket `/dev/log`, from an Internet domain socket specified in `/etc/services`, and from the special device `/dev/klog` (to read kernel messages).

`Syslogd` configures when it starts up and whenever it receives a hangup signal. Lines in the configuration file have a *selector* to determine the message priorities to which the line applies and an *action*. The *action* field are separated from the selector by one or more tabs.

Selectors are semicolon separated lists of priority specifiers. Each priority has a *facility* describing the part of the system that generated the message, a dot, and a *level* indicating the severity of the message. Symbolic names may be used. An asterisk selects all facilities. All messages of the specified level or higher (greater severity) are selected. More than one facility may be selected using commas to separate them. For example:

```
*.emerg;mail,daemon.crit
```

Selects all facilities at the *emerg* level and the *mail* and *daemon* facilities at the *crit* level.

Known facilities and levels recognized by `syslogd` are those listed in `syslog`(3) without the leading "LOG_". The additional facility "mark" has a message at priority LOG_INFO sent to it every 20 minutes (this may be changed with the `-m` flag). The "mark" facility is not enabled by a facility field containing an asterisk. The level "none" may be used to disable a particular facility. For example,

```
*.debug;mail.none
```

Sends all messages *except* mail messages to the selected file.

The second part of each line describes where the message is to be logged if this line is selected. There are four forms:

- A filename (beginning with a leading slash). The file will be opened in append mode.
- A hostname preceeded by an at sign ("@"). Selected messages are forwarded to the `syslogd` on the named host.
- A comma separated list of users. Selected messages are written to those users if they are logged in.
- An asterisk. Selected messages are written to all logged-in users.

Blank lines and lines beginning with '#' are ignored.

For example, the configuration file:

```
kern,mark.debug         /dev/console
*.notice;mail.info      /usr/spool/adm/syslog
*.crit                  /usr/adm/critical
kern.err                @ucbarpa
```

```
*.emerg                 *
*.alert                 eric,kridle
*.alert;auth.warning    ralph
```

logs all kernel messages and 20 minute marks onto the system console, all notice (or higher) level messages and all mail system messages except debug messages into the file /usr/spool/adm/syslog, and all critical messages into /usr/adm/critical; kernel messages of error severity or higher are forwarded to ucbarpa. All users will be informed of any emergency messages, the users "eric" and "kridle" will be informed of any alert messages, and the user "ralph" will be informed of any alert message, or any warning message (or higher) from the authorization system.

The flags are:

`-f` Specify an alternate configuration file.

`-m` Select the number of minutes between mark messages.

`-d` Turn on debugging.

`Syslogd` creates the file `/etc/syslog.pid`, if possible, containing a single line with its process id. This can be used to kill or reconfigure `syslogd`.

To bring `syslogd` down, it should be sent a terminate signal (e.g. `kill ` `cat /etc/syslog.pid` `).

FILES

`/etc/syslog.conf`	the configuration file
`/etc/syslog.pid`	the process id
`/dev/log`	Name of the UNIX domain datagram log socket
`/dev/klog`	The kernel log device

SEE ALSO

`logger`(1), `syslog`(3C), `syslog.conf`(5).

NAME

`systemid` – display the unique system identifier

SYNOPSIS

`/etc/systemid`

DESCRIPTION

`Systemid` prints the vendor stamp and the system identifier. The vendor stamp is a number that uniquely identifies a computer or software manufacturer. The system identifier is a number that uniquely identifies the computer system within those systems manufactured by the indicated vendor. The vendor stamp is displayed as a decimal number on the first line followed by a parenthetical description of the vendor, if the vendor stamp is recognized. The system identifier is displayed as a hexadecimal number on the second line.

EXIT CODES

1 The system identifier could not be determined.

0 The system identifier is available.

EXAMPLES

```
$ systemid
Vendor stamp: 512 (Data General DG/UX)
System id:    012345abc
```

Prints the vendor stamp and the system identifier.

DIAGNOSTICS

"not available" is printed instead of the system identifier if the system identifier cannot be determined.

SEE ALSO

`id`(1), `machid`(1), `uname`(1), `dg_sys_info`(2).

NOTES

The vendor stamp for Data General is `512`.

NAME

tcload – load terminal controller devices

SYNOPSIS

/usr/sbin/tcload [-v] *controller_node* [...] or
/usr/sbin/tcload [-v] -a

DESCRIPTION

The tcload command makes asynchronous terminals (/dev/tty* nodes) available for use by initializing and enabling the desired terminal controller devices in the appropriate way. For instance, for a syac device, tcload resets the board, downloads its resident controller code, starts it executing, monitors events on the device, and responds appropriately to them. The controllers to be loaded are specified by the controller_node arguments.

Note that the tcload process does *not* terminate as in pre-5.4 revisions of DG/UX. It continues to run and react to events on the system's controllers.

The following options are available:

-v Print verbose output about the loading process.

-a Locate and load all terminal controllers configured into the system; this is equivalent to running the command:

```
/usr/sbin/tcload /dev/async/*\)
```

EXAMPLE

```
/usr/sbin/tcload -v /dev/async/0 '/dev/async/syac@61(60020000)'
```

FILES

/dev/async/*	Terminal controller device files
/usr/lib/tcload/models/*	Directories for various controller types
/usr/lib/tcload/models/*/LOAD	Load programs for controller types

SEE ALSO

syac(7).

NAME

telnetd - DARPA TELNET protocol server

SYNOPSIS

telnetd

DESCRIPTION

telnetd is a server which supports the DARPA standard TELNET virtual terminal protocol. telnetd is invoked by the internet server [see inetd(1M)], normally for requests to connect to the TELNET port as indicated by the /etc/services file [see services(4)].

telnetd operates by allocating a pseudo-terminal device for a client, then creating a login process which has the slave side of the pseudo-terminal as its standard input, output, and error. telnetd manipulates the master side of the pseudo-terminal, implementing the TELNET protocol and passing characters between the remote client and the login process.

When a TELNET session is started up, telnetd sends TELNET options to the client side indicating a willingness to do *remote echo* of characters, to *suppress go ahead*, and to receive *terminal type* and *window size information* from the remote client. If the remote client is willing, the remote terminal type is propagated in the environment of the created login process and the X window size is placed in its stty settings.

telnetd is willing to do: *echo*, *binary*, *suppress go ahead*, and *timing mark*. telnetd is willing to have the remote client do: *binary*, *terminal type*, and *suppress go ahead*.

SEE ALSO

telnet(1C).

Postel, Jon, and Joyce Reynolds, "Telnet Protocol Specification," RFC 854, Network Information Center, SRI International, Menlo Park, Calif., May 1983.

NOTES

telnetd has been reimplemented in STREAMS for improved performance. This means that TELNET must be configured into the kernel through the system file resulting in a /dev/telnet file. Also, the /etc/inetd.conf entry for TELNET must have "tli" in the type column, not "stream" [see inetd.conf(4)].

Some TELNET commands are only partially implemented.

The TELNET protocol uses the Negotiate About Window Size protocol sequence to allow for the exchange of the number of lines and columns on the user's terminal.

Binary mode has no common interpretation except between similar operating systems.

The terminal type name received from the remote client is converted to lower case.

telnetd never sends TELNET *go ahead* commands.

NAME

`testlocale` – test locale definition

SYNOPSIS

`testlocale` *localename*

where:

localename

is the name of the locale definition to examine. *localename* should be the name of a directory below `/usr/lib/locale`.

DESCRIPTION

The `testlocale` command endeavors to access all aspects of the specified locale definition, in order to detect any part that may be missing or incorrect.

FILES

The files examined are those that are described in `setlocale`(3C).

SEE ALSO

`chrtbl`(1), `colltbl`(1), `montbl`(1), `setlocale`(3C).

NAME

tftpd – Trivial File Transfer Protocol server

SYNOPSIS

tftpd [-d]

DESCRIPTION

The tftpd server supports the DARPA Trivial File Transfer Protocol (TFTP). The TFTP server is invoked by the inetd server when an incoming connection is detected on the port specified in /etc/services. See inetd(1M) and services(4) for details. If you specify the -d option, each socket created by tftpd will have debugging enabled, with output going to /tmp/tftpd*.

Using tftpd does not require an account or password on the remote system. Due to the lack of authentication information, tftpd allows only publicly readable files to be accessed. Note that this extends the concept of "public" to include all users on all hosts that can be reached through the network; this may not be appropriate on all systems, and its implications should be considered before enabling tftp service. In short, enabling the tftpd daemon on your system creates a potential security problem.

However, you must run tftpd on an OS server to boot a diskless client over the network. Thoroughly check and possibly change the read permissions of sensitive files on your system.

SEE ALSO

inetd(1M), tftp(1C), services(4).

BUGS

The search permissions of the directories leading to the files accessed are not checked.

NAME

tic - TERMINFO compiler

SYNOPSIS

tic [-c] [-v[*n*]] *file*

where:

n An integer from 1 to 10 inclusive, indicating the level of detail desired

file The path to a terminfo(4) source file

DESCRIPTION

The tic command translates a TERMINFO *file* from a textual source format into a binary compiled format. The compiled format is necessary for use with the library routines described in curses(3X). The single *file* argument specifies the pathname of a file containing one or more TERMINFO terminal descriptions in source format (see terminfo(4)). Each description in the file describes the capabilities of a particular terminal.

By default, the resulting binary files are placed under the directory /usr/share/lib/terminfo. However, if the environment variable TERMINFO is set, the compiled results are placed under the directory specified by the value of that variable.

When a use=*entry-name* field is discovered in a terminal entry currently being compiled, tic reads in the binary *entry-name* file to complete the entry. Tic duplicates the capabilities in *entry-name* for the current entry, with the exception of those capabilities that are explicitly defined in the current entry. If *entry-name* is created within the same source *file*, tic will compile and write out *entry-name* before reading it back in to satisfy the use=*entry-name* field. This ordering is guaranteed regardless of the organization of the source *file*; tic performs dependency analysis on the definitions in *file*, and compiles entries in an order that satisfies the use= dependencies. If *entry-name* is not found within *file*, tic will first search the directory named by the environment variable TERMINFO, if that variable is set. If *entry-name* is not found there (or if TERMINFO is not set), tic will then search the default directory /usr/share/lib/terminfo.

Options are:

-v*n* specify that (verbose) trace information be written to standard error showing tic's progress. If *n* is omitted, the default level is 1. If *n* is specified and greater than 1, the level of detail is increased.

-c check *file* for errors only; do not write out compiled entries. Errors in use= links are not detected.

FILES

/usr/share/lib/terminfo/?/*
compiled terminal description data base

DIAGNOSTICS

Parse Errors

All parser error messages are prefixed by the approximate line number of the error and the title of the entry being processed.

File does not start with terminal names in column one

The first line encountered in the file, after comments, must be a list of terminal names.

`'termname': bad first term name.`
or
`'termname': bad term name found in list.`
or
`Illegal terminal name - 'termname'`
`Terminal names must start with a letter or digit,`

The terminal name *termname* was invalid. Names must not contain white space or slashes, and must begin with a letter or digit.

`'termname': terminal name too long.`

An extremely long terminal name *termname* was found.

`'termname': terminal name too short.`

A one-character terminal name *termname* was found.

`At least one synonym should begin with a letter.`

None of the names for the terminal began with a letter.

`Newline in middle of terminal name`

The trailing comma was probably left off of the list of names.

termname`: bad term name`

The current entry specifies an invalid terminal name in the `use=`*termname* field.

`'termname' defined in more than one entry.`

An entry *termname* was defined more than once in the same source file.

`Terminal name 'termname' synonym for itself`

A terminal name *termname* was also listed in its list of synonyms.

`Unknown Capability - 'capname'`

The capability *capname* was found within the file but is not a recognized `terminfo`(4) variable.

`Wrong type used for capability 'capname'`

A boolean capability was given a value, a numeric capability was given a string value, or a string capability was given a numeric value.

`Unknown token type`

The character following a capability was not one of the four allowed: `,` for boolean capabilities, `#` for numeric capabilities, `=` for string capabilities, or `@` to cancel any type of capability.

`Illegal character - 'char'`

The invalid character *char* was found in the input file.

`Illegal ^ character - 'char'`

The second character of a control sequence specification `^`*char* was not a printable ASCII character.

`Illegal character in \ sequence - 'char'`

The second character of a backslash escape sequence `\`*char* did not result in a recognized `terminfo`(4) special sequence.

`Missing comma`

A comma was missing.

`Missing numeric value`

A numeric capability was not followed by any number.

`NULL string value`

A string capability was not followed by any string. (The proper way to delete a string capability is to use the cancel token @.)

`Very long string found. Missing comma?`

A string capability has a very long value. This is usually caused by a missing comma.

`Premature EOF`

The current entry ended prematurely, indicating a syntax error.

`Premature EOF - missing comma?`

The current entry contained an unterminated string capability. This is usually caused by a missing comma.

`Token after a seek not NAMES`

The file being compiled was changed during the compilation.

`Backspaced off beginning of line`

An internal error in `tic` has been detected.

`Error in following up use-links.`

A `use=`*name* capability either referenced a non-existent terminal called *name*, or *name* somehow referred back to the current entry. A list of entries involved is printed following the error message.

Other Errors

A command line error results in a self-explanatory message identifying the problem, followed by a usage summary for the `tic` command. A system call failure results in a message identifying the system operation that failed, as well as the approximate line number and the title of the entry being processed, where appropriate.

SEE ALSO

`captoinfo`(1M), `infocmp`(1M), `curses`(3X), `terminfo`(4), `term`(5).

NOTES

Compiled entries cannot exceed 4096 bytes. The name field cannot exceed 128 bytes.

When the `-c` option is used, duplicate terminal names will not be diagnosed; however, when `-c` is not used, they will be.

When an entry, for instance *entry_name_1*, contains a `use=`*entry_name_2* field, any canceled capabilities in *entry_name_2* must also appear in *entry_name_1* before the `use=` field, if these capabilities are to be canceled in *entry_name_1*.

NAME

trap_recv – receive SNMP trap messages

SYNOPSIS

trap_recv [-d] [*port*]

where:

port is an optional UDP port number

DESCRIPTION

Use the trap_recv command to receive Trap-PDUs from SNMP agents. The command listens on UDP port number *port* for incoming traps and prints information about them as they are received. If you do not specify a *port*, trap_recv will determine the port number using getservbyname(3N) for "snmp-trap". Superuser access is required for any port number less than 1024, including the default port of 162. Use the -d option to print additional information about the packets as they are received.

EXAMPLE

In the following example, trap_recv is used to demonstrate the trap generated by trap_send.

```
$ trap_recv &
$ trap_send localhost public coldStart
Packet Dump:
30 32 02 01 00 04 06 70 75 62 6c 69 63 a4 25 06
09 2b 06 01 03 01 2a 2a 2a 2a 40 04 80 de 08 1e
02 01 00 02 01 00 43 01 00 30 09 30 07 06 03 29
01 01 05 00
Community: public
Enterprise: DataGeneral
Agent-addr: 127.0.0.1
Cold start trap.
Time Ticks: 0
Name: iso.1.1.1
Value: NULL
```

DIAGNOSTICS

Exit status of –1 is returned if there are command line errors or the user does not have access to bind to the UDP port.

SEE ALSO

getmany(1M), getnext(1M), setany(1M), snmpd(1M), trap_send(1M), getservbyname(3N), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

trap_send – send SNMP trap message

SYNOPSIS

trap_send *host community type* [*port*]

where:

host is a hostname or Internet address
community is a community string
type is the trap type
port is an optional UDP port number to send the trap to.

DESCRIPTION

Use the trap_send command to send Trap-PDUs to network management stations (NMS). The trap_send command sends an SNMP message containing the *community* string and a Trap-PDU specifying trap *type* to the NMS at UDP port number *port* running on *host*.

Specify the *host* as either a hostname or an Internet address in dot-notation.

Specify the *community* name as a text string to be used by the NMS to authenticate the request.

Specify the trap *type* as either an integer value or as a text string. Legal values for type are: 1=coldStart, 2=warmStart, 3=linkDown, 4=linkUp, 5=authenticationFailure, 6=egpNeighborLoss, and 7=enterpriseSpecific.

EXAMPLE

In the following example trap_recv is used to demonstrate the trap generated by trap_send.

```
$ trap_recv &
$ trap_send localhost public coldStart
Packet Dump:
30 32 02 01 00 04 06 70 75 62 6c 69 63 a4 25 06
09 2b 06 01 03 01 2a 2a 2a 2a 40 04 80 de 08 1e
02 01 00 02 01 00 43 01 00 30 09 30 07 06 03 29
01 01 05 00
Community: public
Enterprise: experimental.1.42.42.42.42
Agent-addr: 127.0.0.1
Cold start trap.
Time Ticks: 0
Name: iso.1.1.1
Value: NULL
```

DIAGNOSTICS

Exit status is 0 for success. Exit status is –1 if there are errors parsing the command line.

SEE ALSO

getmany(1M), getnext(1M), setany(1M), snmpd(1M), trap_recv(1M), snmpd.communities(4M), snmpd.config(4M), snmpd.trap_communities(4M).

NAME

ttyadm - format and output TTY port monitor information

SYNOPSIS

```
/usr/sbin/ttyadm [ -b ] [ -c ] [ -r count ] [ -h ]
[ -i msg ] [ -m modules ] [ -p prompt ] [ -t timeout ]
-d device -l ttylabel -s service

/usr/sbin/ttyadm -V
```

where:

count	is the prompt delay, an integer character count
msg	is the "disabled" message, a character string
modules	is a comma-separated list of STREAMS modules
prompt	is a character string specifying the TTY prompt
timeout	is the time to wait in seconds
device	is the path to a character-special device file
ttylabel	is an entry in the TTY settings file /etc/ttydefs
service	is the command line for a port service program

DESCRIPTION

The ttyadm command is an administrative command that formats information specific to ttymon(1M) and writes it to the standard output. The Service Access Facility (SAF) requires each port monitor to provide such a command. Note that the port monitor administrative file is updated by the Service Access Controller (sac(1M)) administrative commands, sacadm(1M) and pmadm(1M). ttyadm provides a means of presenting formatted port monitor-specific (i.e., ttymon-specific) data to these commands. (See the NOTE section for higher-level management interfaces.)

Options are:

-b	Set the "bidirectional port" flag. When this flag is set, the line can be used in both directions. ttymon(1M) will allow users to connect to the service associated with the port, but if the port is free, uucico(1M), cu(1), or ct(1) can use it for dialing out.
-c	Set the "connect-on-carrier" flag for the port. If the -c flag is set, ttymon(1M) will invoke the port's associated service immediately when a connect indication is received (i.e., no prompt is printed and no baud-rate searching is done).
-d *device*	Specify the device node for the TTY port. *device* is the full pathname of a character-special device file.
-h	Set the hangup flag for the port. If the -h flag is not set, ttymon(1M) will force a hangup on the line by setting the speed to zero before setting the speed to the default or specified value.
-i *message*	Specify the inactive (disabled) response message. This message will be sent to the TTY port if the port is disabled or the ttymon(1M) monitoring the port is disabled.
-l *ttylabel*	Specify the *ttylabel* in the TTY settings file (/etc/ttydefs) to use as the starting point when searching for the proper baud rate.
-m *modules*	Specify a list of pushable STREAMS modules. The modules will be pushed, in the order in which they are specified (leftmost first), before the service is invoked. *modules* must be a comma-separated list of STREAMS modules, with no white space included. Any modules

currently on the stream will be popped before these modules are pushed.

`-r` *count* Specify the delay before printing a prompt. When the `-r` option is used, `ttymon`(1M) will wait until it receives data from the port before it displays a prompt. If *count* is equal to zero, `ttymon`(1M) will wait until it receives any character. If *count* is greater than zero, `ttymon`(1M) will wait until *count* New Line characters have been received.

`-p` *prompt* Specify the prompt message that `ttymon`(1M) will display. If this option is not given, then `ttyadm` uses the default value "login: ".

`-s` *service* Specify the service to be invoked. *service* is the full pathname of the program to be invoked when a connection request is received. If arguments are required, the command and its arguments must be enclosed in double quotes.

`-t` *timeout* Specify the port timeout. `ttymon`(1M) should close the port if the open on the port succeeds and no input data is received in *timeout* seconds.

`-V` Display the version number of the current `/usr/lib/saf/ttymon` command. This option is mutually exclusive with all other options.

FILES

`/etc/ttydefs`
TTY settings file

DIAGNOSTICS

If successful, `ttyadm` will generate the requested information, write it on the standard output, and exit with a status of `0`. If `ttyadm` is invoked with an invalid number of arguments or invalid arguments, or if an incomplete option is specified, an error message will be written to the standard error and `ttyadm` will exit with a non-zero status.

SEE ALSO

`ct`(1), `cu`(1), `admportservice`(1M), `admterminal`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `sttydefs`(1M), `ttymon`(1M), `uucico`(1M).

System Administrator's Guide, "The Port Monitor `ttymon`."

NOTE

`ttyadm` is a low-level interface to `ttymon`(1M) port and service management. For a high-level, user-friendly interface, use the "Port Services" or "Terminals" menu of `sysadm`(1M) or `xsysadm`(1M). For an intermediate-level, object-oriented interface, use the `admportservice`(1M) or the `admterminal`(1M) utility.

NAME

ttymon – monitor terminal ports

SYNOPSIS

/usr/lib/saf/ttymon

/usr/lib/saf/ttymon -g [-d *device*] [-h] [-t *timeout*] [-l *ttylabel*] [-p *prompt*] [-m *modules*] [-T *term-variable*]

where:

device is the pathname to a character-special device file
timeout is an integral number of seconds to wait for input
ttylabel is a name in the TTY settings file /etc/ttydefs
prompt is a string to be output to the *device*
modules is a comma-separated list of STREAMS modules
term-variable
is the value to which the TERM environment variable should be set for the device.

DESCRIPTION

ttymon is a STREAMS-based TTY port monitor. It monitors device ports, sets terminal modes, baud rates, and line disciplines for the ports, and connects users or applications to services associated with the ports. Normally, ttymon is configured to run under the Service Access Controller, sac(1M), as part of the Service Access Facility (SAF). It is managed using the sacadm(1M) command. Each instance of ttymon can monitor multiple ports. The ports monitored by an instance of ttymon are specified in the port monitor's administrative file. The administrative file is managed using the pmadm(1M) and ttyadm(1M) commands. (See the NOTES section for higher-level management interfaces.)

When an instance of ttymon is invoked by the sac(1M) command, it starts to monitor its ports. For each port, ttymon first initializes the line disciplines, if they are specified, followed by the speed and terminal settings. The values used for initialization are taken from the appropriate entry in the TTY settings file. This file is maintained by the sttydefs(1M) command. Default line disciplines on ports are usually set up by the autopush(1M) command, interfacing with the STREAMS Autopush Driver (SAD).

ttymon then writes the prompt and waits for user input. If the user indicates that the speed is inappropriate by pressing the BREAK key, ttymon tries the next speed listed in the TTY settings file and writes the prompt again. When valid input is received, ttymon interprets the per-service configuration file for the port, if one exists, creates a utmp(4) entry if required, establishes the service environment, and then invokes the service associated with the port. Valid input consists of a string of at least one character (other than New Line), terminated by a New Line. After the service terminates, ttymon cleans up the utmp(4) entry, if one exists, and returns the port to its initial state.

If the "autobaud" feature is enabled for a port, ttymon will try to determine the baud rate on the port automatically. Users must enter a carriage return before ttymon can recognize the baud rate and print the prompt. Currently, the baud rates that can be determined by this feature are 110, 1200, 2400, 4800, and 9600 baud.

If a port is configured as a bidirectional port, ttymon will allow users to connect to a service, and, if the port is free, will allow uucico(1M), cu(1), or ct(1) to use it for dialing out. If a port is bidirectional, ttymon will wait to read a character before it prints a prompt.

If the "connect-on-carrier" flag (see ttyadm(1M)) is set for a port, ttymon will immediately invoke the port's associated service when a connection request is received. The prompt message will not be sent.

If a port is disabled, ttymon will not start any service on that port. If a disabled message is specified, ttymon will send out the message when a connection request is received. If ttymon is disabled, all ports under that instance of ttymon will also be disabled.

Service Invocation

The service ttymon invokes for a port is specified in the ttymon administrative file. ttymon will scan the character string giving the service to be invoked for this port, looking for a %d or a %% two-character sequence. If %d is found, ttymon will modify the service command string by replacing those two characters with the full path name of this port (the device name). If %% is found, those two characters will be replaced by a single % character.

When the service is invoked, file descriptors 0, 1, and 2 are opened to the port device for reading and writing. The service is invoked with the user ID, group ID, and current home directory set to that of the user name under which the service was registered with ttymon. Two environment variables, HOME and TTYPROMPT, are added to the service's environment by ttymon. HOME is set to the home directory of the user name under which the service is invoked. TTYPROMPT is set to the prompt string configured for the service on the port. This is provided so that a service invoked by ttymon has a means of determining if a prompt was actually issued by ttymon and, if so, what that prompt actually was.

See ttyadm(1M) for options that can be set for ports monitored by ttymon under the Service Access Controller.

Stand-Alone Operation

A special invocation of ttymon is provided that runs independently of the SAC (i.e., does not depend on sac(1M) or its configuration files). This independent "express" mode is the only invocation of ttymon that accepts command line options and arguments. This form of the command should be called only by applications that need to set the correct baud rate and terminal settings on a port and then connect to login(1) service, but that cannot be pre-configured under the SAC.

Options are:

-g Enable the independent "express" mode of ttymon. This option is required in order to use any of the other options below.

-d *device*
Attach ttymon to the port *device*, which must be the full path name of a character-special device file. If this option is not specified, the invoking process must connect file descriptor 0 to a TTY port before starting ttymon.

-h Do not disconnect the line before initializing the terminal settings. If this option is not set, ttymon will force a hangup on the line by setting the speed to zero before setting the speed to the default or specified speed.

-t *timeout*
Specify that ttymon should exit if no one types anything in *timeout* seconds after the prompt is sent.

-l *ttylabel*
Look up the entry named *ttylabel* in the /etc/ttydefs file. This definition tells ttymon at what speed to run initially, what the initial TTY settings are,

and what speed to try next if the user indicates that the speed is inappropriate by pressing the BREAK key. The default speed is 9600 baud.

`-p` *prompt*
Print the prompt string *prompt* when the port is ready for input. The default prompt is "Login: ".

`-m` *modules*
When initializing the port, pop all modules on the port, and then push *modules* in the order specified (leftmost first). *modules* is a comma-separated list of pushable STREAMS modules. Default modules on the ports are usually set up by the `autopush`(1M) command, interfacing with the STREAMS Autopush Driver (SAD).

`-T` *term-variable*
Set the TERM environment variable to *term-variable* for the device.

EXAMPLES

```
/usr/lib/saf/ttymon -g -d /dev/console -m ldterm -l console
```

This example runs `ttymon` in express mode on the console device, `/dev/console`. The basic STREAMS line discipline module `ldterm` will be pushed onto the console stream, and the `console` entry in the TTY definitions file will be referenced to set the line speed and terminal settings.

FILES

`/etc/ttydefs`
TTY settings file

SEE ALSO

`ct`(1), `cu`(1), `login`(1), `admportservice`(1M), `admterminal`(1M), `autopush`(1M), `pmadm`(1M), `sac`(1M), `sacadm`(1M), `sttydefs`(1M), `sysadm`(1M), `ttyadm`(1M), `uucico`(1M), `utmp`(4).

System Administrator's Guide, "The Port Monitor `ttymon`."

NOTES

If a port is monitored by more than one `ttymon`, it is possible for the `ttymons` to send out prompt messages in such a way that they compete for input.

For a high-level, user-friendly interface to `ttymon` and port services, use the "Ports" menu of `sysadm`(1M) or `xsysadm`(1M). For an intermediate-level, object-oriented interface, use the `admportmonitor`(1M) and `admportservice`(1M) utilities.

NAME

`tunefs` – tune an existing file system

SYNOPSIS

`/etc/tunefs` [*option* ...] *filesys*|*special*

where:

option	`-e`, `-E`, `-m`, `-s`, `-S`, `-e`, `-x`, or `-X`, followed by an argument.
filesys	The pathname of the directory with which the file system is associated in the file `/etc/fstab`
special	The pathname of a special file referring to a device containing a file system

DESCRIPTION

`Tunefs` changes the dynamic parameters of a file system that affect the layout policies. Defaults are set according to the values established by `mkfs`(1M). The file system must be unmounted.

Options are:

`-e` *max_blks_first_dar*	Specifies that when allocating disk space for a file, *max_blks_first_dar* disk blocks should be allocated from the disk allocation region (DAR) chosen for the initial file space allocation, before moving on to another DAR. The valid range for *max_blks_first_dar* is from 1 to the number of available disk blocks in the DAR.
`-E` *max_blks_per_dar*	Specifies that when allocating disk space for a file, after the initial disk allocation region (DAR) has been exhausted (see the `-e` option), then *max.blocks_per_dar* disk blocks should be allocated from each subsequent DAR. The valid range for *max.blks_per_dar* is from *max_blocks_first_dar* (see the `-e` option) to the number of available disk blocks in the DAR.
`-m` *min_free*	Specifies the minimum percentage of the file system that should be maintained as free space. Only superuser processes may allocate these blocks. If *min_free* is greater than the percentage of space currently free, then non-superuser processes will not be able to allocate any blocks on the file system until the percentage free reaches *min_free*. The valid range for *min_free* is from 1 to 100 percent; however, the disk I/O performance may degrade as the amount of free space approaches 0 percent.
`-s` *max_data*	Specifies the block size for a file's data blocks on the file system, given as log base 2 of the block size. The disk space for a file's data is allocated 2^*max_data* blocks at a time. The valid range for *max_data* is from 1 to the log of the number of user blocks per disk allocation region in the file system.
`-S` *max_dir*	Specifies the block size for directories on the file system, given as log base 2 of the block size. The disk space for the file system's directory blocks is allocated 2^*max_dir* blocks at at time. The valid range for *max_dir* is from 1 to the log of the number of user blocks per disk allocation region in the file system.

-x *max_idx_data* Specifies the block size for a file's index blocks (the space used for addressing a file's data blocks) in the file system, given as log base 2 of the block size. The disk space for a file's index blocks is allocated 2^*max_idx_data* blocks at a time. The valid range for *max_idx_data* is from 1 to the log of the number of blocks available for use as index blocks per disk allocation region (DAR), which is usually less than the number of user blocks per DAR.

-X *max_idx_dir* Specifies the block size for the file system's index blocks (the space used for addressing a directory's blocks) in the file system, given as log base 2 of the block size. The disk space for a directory's index blocks is allocated 2^*max_idx_dir* blocks at at time. The valid range for *max_idx_dir* is from 1 to the log of the number of blocks available for use as index blocks per disk allocation region (DAR), which is usually less than the number of user blocks per DAR.

Specifying an invalid option value causes the program to print the valid range of values for that option and then exit.

SEE ALSO

`fsck`(1M), `mkfs`(1M), `fstab`(4).

BUGS

`Tunefs` cannot be run on the root file system because that system cannot be unmounted.

NAME

useradd - administer a new user login on the system

SYNOPSIS

useradd [-y] [-u *uid* [-o]] [-g *group*] [-G *group*[,*group*...] [-d *dir*]
[-s *shell*] [-c *comment*] [-m [-k *skel_dir*]] [-f *inactive*]
[-e *expire*] *login*

useradd -D [-g *group*] [-b *base_dir*] [-f *inactive*] [-e *expire*]

DESCRIPTION

Invoking useradd without the -D option adds a new user entry to the /etc/passwd file. It also creates supplementary group memberships for the user (-G option) and creates the home directory (-m option) for the user if requested. The new login remains locked until the passwd(1M) command is executed.

Invoking useradd -D with no additional options displays the default values for *group*, *base_dir*, *shel_dir*, *shell*, *inactive*, and *expire*. The values for *group*, *base_dir*, *inactive*, *expire*, and *shell* are used for invocations without the -D option.

Invoking useradd -D with -g, -b, -f, or -e (or any combination of these) sets the default values for the respective fields. [As installed, the default group is general (group ID of 100) and the default value of *base_dir* is /home]. Subsequent invocations of useradd without the -D option use these arguments.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

The following options are available:

-y Perform the requested operation on the global NIS (YP) database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-u *uid* The UID of the new user. This UID must be a non-negative decimal integer below MAXUID as defined in <sys/param.h>. The UID defaults to the next available (unique) number above the highest number currently assigned. For example, if UIDs 100, 105, and 200 are assigned, the next default UID number will be 201. (UIDs from 0-99 are reserved.)

-o This option allows a UID to be duplicated (non-unique).

-g *group* An existing group's integer ID or character-string name. Without the -D option, it defines the new user's primary group membership and defaults to the default group. You can reset this default value by invoking useradd -D -g *group*.

-G *group* An existing group's integer ID or character-string name. It defines the new user's supplementary group membership. Duplicates between *group* with the -g and -G options are ignored. No more than NGROUPS_MAX groups may be specified.

-d *dir* The home directory of the new user. It defaults to *base_dir*/*login*, where *base_dir* is the base directory for new login home directories and *login* is the new login.

`-s` *shell* Full pathname of the program used as the user's shell on login. It defaults to an empty field causing the system to use `/usr/bin/sh` as the default. The value of *shell* must be a valid executable file.

`-c` *comment* Any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. This information is stored in the user's `/etc/passwd` entry.

`-m` Create the new user's home directory if it doesn't already exist. If the directory already exists, it must have read, write, and execute permissions by *group*, where *group* is the user's primary group.

`-k` *skel_dir* A directory that contains skeleton information (such as `.profile`) that can be copied into a new user's home directory. This directory must exist. The system provides a "skel" directory (`/etc/skel`) that can be used for this purpose.

`-e` *expire* The date on which a login can no longer be used; after this date, no user will be able to access this login. (This option is useful for creating temporary logins.) You may type the value of the argument *expire* (which is a date) in any format you like (except a Julian date). For example, you may enter `10/6/90` or `October 6, 1990`. A value of ``''`` defeats the status of the expired date.

`-f` *inactive* The maximum number of days allowed between uses of a login `ID` before that login `ID` is declared valid. Normal values are positive integers. A value of `-1` defeats the status.

login A string of printable characters that specifies the existing login name of the user. It must exist and may not contain a colon (`:`) or a newline (`\n`).

login A string of printable characters that specifies the new login name of the user. It may not contain a colon (`:`) or a newline (`\n`).

`-b` *base_dir* The default base directory for the system. If `-d` *dir* is not specified. *base_dir* is concatenated with the user's login to define the home directory. If the `-m` option is not used, `base_dir` must exist.

FILES

```
/etc/passwd
/etc/group
/etc/skel
```

DIAGNOSTICS

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

`0` The command was executed successfully.

`2` The command line syntax was invalid. A usage message for the `useradd` command is displayed.

`3` An invalid argument was provided with an option.

`4` The *uid* specified with the `-u` option is already in use.

`6` The *group* specified with the `-g` option does not exist.

`9` The specified *login* is not unique.

`10` Cannot update `/etc/group`. The login was added to the `/etc/passwd` file but not to the `/etc/group` file.

12 Unable to create the home directory (with the `-m` option) or unable to complete the copy of *skel_dir* to the home directory.

SEE ALSO

`groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `listusers`(1), `logins`(1M), `passwd`(1), `passwd`(1M), `userdel`(1M), `usermod`(1M).

NAME

`userdel` - delete a user's login from the system

SYNOPSIS

`userdel` [`-y`] [`-r`] *login*

DESCRIPTION

The `userdel` command deletes a user's login from the system and makes the appropriate login-related changes to the system file and file system.

The following options are available:

`-y` Perform the requested operation on the global NIS (YP) database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The `-y` option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

`-r` Remove the user's home directory from the system. This directory must exist. The files and directories under the home directory will no longer be accessible following successful execution of the command.

`login` A string of printable characters that specifies an existing login on the system. It may not contain a colon (:), or a newline (\n).

FILES

`/etc/passwd`
`/etc/group`

DIAGNOSTICS

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

`0` Success.

`2` Invalid command syntax. A usage message for the `userdel` command is displayed.

`6` The login to be removed does not exist.

`8` The login to be removed is in use.

`10` Cannot update the `/etc/group` file but the login is removed from the `/etc/passwd` file.

`12` Cannot remove or otherwise modify the home directory.

SEE ALSO

`groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `listusers`(1), `logins`(1M), `passwd`(1), `passwd`(1M), `useradd`(1M), `usermod`(1M).

NAME

usermod – modify a user's login information on the system

SYNOPSIS

usermod [-y] [-u *uid* [-o]] [-g *group*] [-G *group*[,*group*...] [-d *dir* [-m]]
[-s *shell*] [-c *comment*] [-l *new_logname*] [-f *inactive*]
[-e *expire*] *login*

DESCRIPTION

The usermod command modifies a user's login definition on the system. It changes the definition of the specified login and makes the appropriate login-related system file and file system changes.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

The following options are available:

-y Perform the requested operation on the global NIS (YP) database. Without this option, the requested operation is performed on the local database in the /etc directory. This option is valid only when the machine on which the command is run is the NIS master. The -y option uses the default source directory derived from the SRC_DIR variable specified in the NIS makefile (/etc/yp/Makefile).

-u *uid* New UID for the user. It must be a non-negative decimal integer below MAXUID as defined in <sys/param.h>.

-o This option allows the specified UID to be duplicated (non-unique).

-g *group*
An existing group's integer ID or character-string name. It redefines the user's primary group membership.

-G *group*
An existing group's integer "ID" "," or character string name. It redefines the user's supplementary group membership. Duplicates between group with the -g and -G options are ignored. No more than NGROUPS_UMAX groups may be specified as defined in <sys/param.h>.

-d *dir* The new home directory of the user. It defaults to *base_dir/login*, where *base_dir* is the base directory for new login home directories, and *login* is the new login.

-m Move the user's home directory to the new directory specified with the -d option. If the directory already exists, it must have permissions read/write/execute by *group*, where *group* is the user's primary group.

-s *shell*
Full pathname of the program that is used as the user's shell on login. The value of *shell* must be a valid executable file.

-c *comment*
Any text string. It is generally a short description of the login, and is currently used as the field for the user's full name. This information is stored in the user's /etc/passwd entry.

-l *new_logname*
A string of printable characters that specifies the new login name for the user. It may not contain a colon (:) or a newline (\n).

`-e` *expire*
: The date on which a login can no longer be used; after this date, no user will be able to access this login. (This option is useful for creating temporary logins.) You may type the value of the argument *expire* (which is a date) in any format you like (except a Julian date). For example, you may enter `10/6/90` or `October 6, 1990`. A value of `` `` '' `` defeats the status of the expired date.

`-f` *inactive*
: The maximum number of days allowed between uses of a login `ID` before that login `ID` is declared valid. Normal values are positive integers. A value of `-1` defeats the status.

login
: A string of printable characters that specifies the existing login name of the user. It must exist and may not contain a colon (`:`), or a newline (`\n`).

FILES

`/etc/passwd`
`/etc/group`

DIAGNOSTICS

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

`0` The command was executed successfully.

`2` The command syntax was invalid. A usage message for the `usermod` command is displayed.

`3` An invalid argument was provided to an option.

`4` The *uid* given with the `-u` option is already in use.

`6` The login to be modified does not exist or *group* does not exist.

`8` The login to be modified is in use.

`9` The *new_logname* is already in use.

`10` Cannot update the `/etc/group` file. Other update requests will be implemented.

`11` Insufficient space to move the home directory (`-m` option). Other update requests will be implemented.

`12` Unable to complete the move of the home directory to the new home directory.

SEE ALSO

`groupadd`(1M), `groupdel`(1M), `groupmod`(1M), `listusers`(1), `logins`(1M), `passwd`(1), `passwd`(1M), `useradd`(1M), `userdel`(1M).

NAME

uucheck - check the uucp directories and permissions file

SYNOPSIS

/usr/lib/uucp/uucheck [-v] [-x *debug_level*]

DESCRIPTION

Uucheck checks for the presence of the uucp system required files and directories. It also checks for some obvious errors in the Permissions file (/etc/uucp/Permissions). When executed with the -v option, it gives a detailed explanation of how the uucp programs will interpret the Permissions file.

If you have a source license, the -x option is valuable for debugging. *Debug-option* is a single digit in the range 1-9; the higher the value, the greater the detail.

Note that uucheck can only be used by the super-user or uucp.

FILES

/etc/uucp/Systems
/etc/uucp/Permissions
/etc/uucp/Devices
/etc/uucp/Maxuuscheds
/etc/uucp/Maxuuxqts
/usr/spool/uucp/*
/usr/spool/locks/LCK*
/usr/spool/uucppublic/*

SEE ALSO

uucico(1M), uusched(1M), uucp(1), uustat(1), uux(1).

NOTE

The program does not check file/directory modes or some errors in the Permissions file, such as duplicate login or machine name.

NAME

uucico - file transport program for the uucp system

SYNOPSIS

/usr/lib/uucp/uucico [-r *role_number*] [-x *debug_level*]
[-i *interface*] [-d *spool_directory*] -s *system_name*

DESCRIPTION

Uucico is the file transport program for uucp work file transfers. Role numbers for the -r are the digit 1 for master mode or 0 for slave mode (default). Programs or cron that start uucico should use -r1. Uux and uucp both queue jobs that will be transferred by uucico. It is normally started by the scheduler, uusched, but can be started manually; this is done for debugging. A single digit in the range 0-9 must be used with the -x option with higher numbers specified for more debugging information.

The -i option defines the interface used with uucico. This interface affects only slave mode. DG/UX is the only known interface at this time and is the default.

The -d option specifies the alternate directory as *spool_directory* rather than /usr/spool/uucp.

Use the -s option to specify the hostname of the remote machine you want uucico to call. Uucico will call this machine even if the spool directory contains no work for that system. This option is useful for polling other machines that do not have the hardware needed to initiate a connection. The machine you specify must be entered into the Systems file.

FILES

/etc/uucp/Systems
/etc/uucp/Permissions
/etc/uucp/Devices
/etc/uucp/Sysfiles
/etc/uucp/Maxuuxqts
/etc/uucp/Maxuuscheds
/usr/spool/uucp/*
/usr/spool/locks/LCK*
/usr/spool/uucppublic/*

SEE ALSO

cron(1M), uucp(1), uusched(1M), uustat(1), uutry(1M), uux(1).

NAME

uucleanup - uucp spool directory clean-up

SYNOPSIS

/usr/lib/uucp/uucleanup [-C*time*] [-W*time*] [-X*time*]
[-m*string*] [-o*time*] [-s*system*]

DESCRIPTION

Uucleanup scans the spool directories for old files and takes appropriate action to remove them in one of the following ways:

- Inform the requester of send/receive requests for systems that can not be reached.
- Return mail, which cannot be delivered, to the sender.
- Delete or execute rnews for rnews type files (depending on where the news originated--locally or remotely). rnews is a commonly used news reading program available in public domain software.
- Remove all other files.

In addition, there is provision to warn users of requests that have been waiting for a given number of days (default 1). Note that uucleanup will process as if all option *times* were specified to the default values unless you specifically set *time* (see **NOTES**).

The following options are available:

-C*time* Any C. files greater or equal to *time* days old will be removed with appropriate information to the requester. (default 7 days)

-D*time* Any D. files greater or equal to *time* days old will be removed. An attempt will be made to deliver mail messages and execute rnews when appropriate. (default 7 days)

-W*time* Any C. files equal to *time* days old will cause a mail message to be sent to the requester warning about the delay in contacting the remote. The message includes the *JOBID*, and in the case of mail, the mail message. The administrator may include a message line telling whom to call to check the problem (-m option). (default 1 day)

-X*time* Any X. files greater or equal to *time* days old will be removed. The D. files are probably not present (if they were, the X. could get executed). But if there are D. files, they will be taken care of by D. processing. (default 2 days)

-m*string* This line will be included in the warning message generated by the -W option. The default line is "See your local administrator to locate the problem".

-o*time* Other files whose age is more than *time* days will be deleted. (default 2 days)

-s*system* Execute for *system* spool directory only.

-x*debug_level*
The -x debug level is a single digit between 0 and 9; higher numbers give more detailed debugging information. (If you have a source license and do not want debugging, compile with the -DSMALL option.)

This program is typically started by the shell uudemon.cleanup, which should be started by cron(1M).

FILES

`/etc/uucp`, `/usr/lib/uucp`
directories with commands used by `uucleanup` internally

`/usr/spool/uucp` spool directory

NOTES

The `-C`, `-D`, `-W`, `-X`, and `-o` options require a value greater than zero.

SEE ALSO

`cron`(1M), `uucp`(1C), `uux`(1C).

NAME

uusched - the scheduler for the uucp file transport program

SYNOPSIS

/usr/lib/uucp/uusched [-x *debug_level*] [-u *debug_level*]

DESCRIPTION

Uusched is the uucp file transport scheduler. It is usually started by the daemon uudemon.hour that is started by cron(1M) from an entry in /usr/spool/cron/crontab:

39 * * * * /bin/su uucp -c "/etc/uucp/uudemon.hour > /dev/null"

The two options are for debugging purposes only: -x debug_level will output debugging messages from uusched and -u *debug_level* will output debugging messages from uucico. The *debug_level* is a number between 0 and 9; higher numbers give more detailed information.

FILES

/etc/uucp/Systems
/etc/uucp/Permissions
/etc/uucp/Devices
/usr/spool/uucp/*
/usr/spool/locks/LCK*
/usr/spool/uucppublic/*

SEE ALSO

cron(1M), uucico(1M), uucp(1), uustat(1), uux(1).

NAME

Uutry - try to contact remote system with debugging on

SYNOPSIS

/usr/lib/uucp/Uutry [-x *debug_level*] [-r] *system_name*

DESCRIPTION

Uutry is a shell that invokes uucico to call a remote site. Debugging is turned on (default is level 5); -x will override that value. The -r overrides the retry time in /usr/spool/uucp/.Status. The debugging output is put in file /tmp/*system_name*. A tail -f of the output is executed. A Ctrl-C will give control back to the terminal while the uucico continues to run, putting its output in /tmp/*system_name*.

FILES

/etc/uucp/Systems
/etc/uucp/Permissions
/etc/uucp/Devices
/etc/uucp/Maxuuxqts
/etc/uucp/Maxuuscheds
/usr/spool/uucp/*
/usr/spool/locks/LCK*
/usr/spool/uucppublic/*
/tmp/system_name

SEE ALSO

uucico(1M), uucp(1), uux(1).

NAME

uuxqt - execute remote command requests

SYNOPSIS

/usr/lib/uucp/uuxqt [-s *system*] [-x *debug_level*]

DESCRIPTION

Uuxqt is the program that executes remote job requests from remote systems generated by the use of the uux command. (Mail uses uux for remote mail requests). uuxqt searches the spool directories looking for X. files. For each X. file, uuxqt checks to see if all the required data files are available and accessible, and file commands are permitted for the requesting system. The Permissions file is used to validate file accessibility and command execution permission.

There are two environment variables that are set before the uuxqt command is executed:
UU_MACHINE is the machine that sent the job (the previous one).
UU_USER is the user that sent the job.
These can be used in writing commands that remote systems can execute to provide information, auditing, or restrictions.

The -x *debug_level* is a single digit between 0 and 9. Higher numbers give more detailed debugging information.

FILES

/etc/uucp/Permissions
/etc/uucp/Maxuuxqts
/usr/spool/uucp/*
/usr/spool/locks/LCK*

SEE ALSO

uucico(1M), uucp(1C), uustat(1C), uux(1C), mail(1).

NAME

vipw – edit the system password file

SYNOPSIS

vipw

DESCRIPTION

vipw edits the password file while setting the appropriate locks, and does any necessary processing after the password file is unlocked. If the password file is already being edited, then you will be told to try again later. The vi editor will be used unless the environment variable EDITOR indicates an alternate editor. vipw performs a number of consistency checks on the password entry for root, and will not allow a password file with a "mangled" root entry to be installed.

FILES

/etc/ptmp

SEE ALSO

admuser(1M), passwd(1), sysadm(1M), passwd(4).

NAME

volcopy, labelit - copy file systems with label checking

SYNOPSIS

/usr/sbin/volcopy [*options*] *fsname special1 volname1 special2 volname2*

/usr/sbin/labelit *special* [*fsname volume* [-n]]

DESCRIPTION

Volcopy makes a literal copy of the file system using a blocksize matched to the device.

Options are:

-a Invoke a verification sequence requiring a positive operator response instead of the standard 10-second delay before the copy is made.

-s (Default) invoke the DEL if wrong verification sequence.

Options used with tapes only are:

-bpi Bits per inch (800 or 1600)

-feet Size of reel in feet (1200 or 2400)

-reel Beginning reel number for a restarted copy

-buf Use double buffered I/O.

Volcopy requests length and density information if it is not given on the command line or is not recorded on an input tape label. If the file system is too large for one reel, volcopy prompts for additional reels. Labels of all reels are checked. You can mount tapes on two or more drives. If volcopy is interrupted, it asks whether you want to quit or to use a shell. In the latter case, you can perform other operations (e.g.,: labelit) and return to volcopy by exiting from the new shell.

The *fsname* argument represents the mounted name (e.g.,: root, u1) of the file system being copied.

The *special* argument is the physical disk section or tape (e.g.,: /dev/dsk/usr or /dev/rmt/0m).

The *volname* is the physical volume name (e.g.,: pk3, t0122) and should match the external label sticker. Such label names are limited to six or fewer characters. If you want to use the existing volume name, specify - for *volname*.

Special1 and *volname1* are the device and volume from which the copy of the file system is being extracted. If *special1* is mounted, fsck must be run on the destination disk file system before that file system can be mounted.

Special2 and *volname2* are the target device and volume. If *special2* is a disk filesystem, it should be unmounted before volcopy is performed.

Labelit can provide initial labels for unmounted disk or tape file systems. With the optional arguments omitted, labelit prints current label values. Otherwise, the tape is relabeled, destroying the previous contents. The -n labels new tapes only, skipping the check of the current label.

EXAMPLES

To copy the root directory into /test , where /test is the file system name associated with the /test file system, no volume label exists, and *fsname* = /, you would do the following:

1) Use labelit to label backup tapes for /test.

```
labelit /dev/rmt/0 /test vol 1
```

2) Create backups with `volcopy` as follows:
`volcopy /test /dev/rdsk/test - /dev/rmt/0 vol 1`

3) Use `umount`(1M) to mount the `/test` directory as follows:
`umount /test`

4) Copy `/` (root directory) to `/test` as follows:
`volcopy / /dev/rdsk/root - /dev/rdsk/test -`

5) Check the `/test` file system for inconsistencies as follows:
`fsck /dev/rdsk/test`

6) Use `labelit`(1M) to verify that the `/test` file system superblock now contains *fsname* = /.

FILES

`/etc/log/filesave.log`
Record of file systems/volumes copied. Must be present in order for `volcopy` to run; an initial file (zero length) is provided with the system.

SEE ALSO

`sh`(1), `fs`(4).

NOTES

Only device names beginning with `/dev/rmt/` are treated as tapes.

`Volcopy` overwrites all of *special2* including the superblock. Any data previously found on that file system will be lost once `volcopy` is performed. If you use volcopy for disk-to-disk transfers, both file systems should be the same size.

NAME

vsccheck – verify that the VSC synchronous controller is operable

SYNOPSIS

vsccheck *board_number*

where:

board_number A two-digit number containing a leading zero if necessary

DESCRIPTION

Vsccheck sends a command to the board resident software that verifies that the VSC controller is operable. The VSC controller is considered to be operable if the board resident software has been downloaded and is capable of performing DMA operations across the VME bus.

The only parameter for this command is the board number of the VSC controller to be checked. The first controller is 01.

EXAMPLES

The following example checks VSC controller 3:

```
% vsccheck 03
        vsccheck: VSC controller 03 is operable
```

DIAGNOSTICS

A zero status will be returned if the command succeeds. A non-zero return status message will be returned if the VSC controller is not operational. In both cases, an informative message describing the operability of the controller will be written to standard error.

SEE ALSO

vscload(1M), ssid(7).

NAME

vscload - download board resident software onto VSC synchronous controller

SYNOPSIS

vscload [-v] [-r] [-e] [-m] -f *image* -p *task_priority* -t *task_id* *device*|-a

where:

task_priority	Task priority, an integer in the range 5 to 255.
task_id	Task identifier, an integer in the range 5 to 255.
device	The direct ssid access device for the VSC controller to be loaded. This option is mutually exclusive with the -a option.

DESCRIPTION

Vscload downloads the board resident software onto the VSC synchronous controller. Only super user is allowed to use this command.

Options are:

-v Specify verbose mode. If verbose is specified, extra information about the load (such as the starting address) is displayed on the console.

-r Force a software reset of the synchronous card prior to the load. This option must be specified only for the first image to be downloaded. In other words, the first image to be downloaded must have this option and this option must not appear for any subsequent loads.

-e Inform the ssid driver that this is the last image to download to the VSC synchronous controller. When ssid receives this command it will assume that the VSC controller is completely loaded and communications can begin.

-m Specify that the download image is the board manager. All messages sent to on board user tasks will be sent directly to the board manager task for on board message routing. It is the responsibility of the board manager to see that all messages are posted to the correct destination task.

-f *image*
Specify the name of the image that is to be downloaded. The image is a binary representation of a hexadecimal file. The image parameter must immediately follow the -f option.

-p *task_priority*
Set the task priority on the synchronous controller card for the image to be loaded. The lower the number, the higher the priority.

-t *task_id*
Set the id of the downloaded task on the VSC. These task ids are used for on board intertask communications.

-a Make vscload search for all VSC controllers configured for ssid, then send the image and related commands to each board automatically.

EXAMPLES

In the following example we will download three images into the first VSC controller configured:

```
vscload -r -v -f vscinit.bin -t 5 -p 5 /dev/ssid01
vscload -v -f vscd.bin -t 6 -p 6 /dev/ssid01
vscload -e -v -m -f vscmgr.bin -t 7 -p 7 /dev/ssid01
```

Note that we reset the VSC prior to loading the first image by using the -r option on

the first vscload command line. We also identified `vscmgr.bin` as being the VSC board manager by specifying the `-m` option on the command line used to load the vscmgr image. We specified the end of the load sequence by the `-e` option on the command line of the last image downloaded.

SEE ALSO

`vsccheck`(1M), `ssid`(7).

NOTES

If you invoke `vscload` manually, you must reset the VSC controller on the first image downloaded with the `-r` option. On the last image, the `-e` option must be specified in order to put the `ssid` driver into the proper state.

NAME

wall – write to all users

SYNOPSIS

/usr/sbin/wall

DESCRIPTION

Wall reads its standard input until an end-of-file. It then sends a message to all currently logged-in users preceded by:

Broadcast Message from . . .

Wall is typically used to warn all users prior to shutting down the system.

The sender must be superuser to override any protections the users may have invoked (see mesg(1)).

International Features

wall can send characters from supplementary codesets.

wall uses the locale of the sender to determine printability.

FILES

/dev/tty*

DIAGNOSTICS

``Cannot send to ...''
when the open on a user's *tty* file fails.

SEE ALSO

mesg(1), write(1).

NAME

wchrtbl – generate character classification and conversion tables

SYNOPSIS

wchrtbl [*file*]

DESCRIPTION

Wchrtbl creates tables containing information on character classification, character conversion, character set width and numeric editing for ASCII and supplementary code sets. The first table is a byte-sized array encoded such that a table lookup can be used to determine the character classification of a character, convert a character [see ctype(3C) and wctype(3W)] and find the byte and screen width of a character in one of the supplementary code sets. The size of the array is (257*2) + 7 bytes: 257 bytes are required for the 8-bit code set character classification table, 257 bytes for the upper- to lower-case and lower- to upper-case conversion table, and 7 bytes for character set width information. The second table is 2 bytes long and is encoded such that the first byte is used to specify the decimal delimiter and the second byte the thousand delimiter. If supplementary code sets are specified, additional variable sized tables are generated for multibyte character classification and conversion.

wchrtbl reads the user-defined character classification and conversion information from *file* and creates three output files in the current directory. One output file, wctype.c (a C-language source file), contains the variable sized array generated from processing the information from *file*. You should review the content of wctype.c to verify that the array is set up as you had planned. The first 257 bytes of the array in wctype.c are used for character classification for single byte characters. The characters used for initializing these bytes of the array represent character classifications that are defined in /usr/include/ctype.h; for example, _L means a character is lower case and _S|_B means the character is both a spacing character and a blank. The second 257 bytes of the array are used for character conversion. These bytes of the array are initialized so that characters for which you do not provide conversion information will be converted to themselves. When you do provide conversion information, the first value of the pair is stored where the second one would be stored normally, and vice versa; for example, if you provide <0x41 0x61>, then 0x61 is stored where 0x41 would be stored normally, and 0x61 is stored where 0x41 would be stored normally. The last 7 bytes are used for character width information. Up to three supplementary code sets can be specified.

For supplementary code sets, there are three sets of tables. The first set is three pointer arrays which point to supplementary code set information tables. If the corresponding supplementary code set information is not specified, the contents of the pointers are zeros. The second one is a set of three supplementary code set information tables. Each table contains minimum and maximum code values to be classified and converted, and also contains pointers to character classification and conversion tables. If there is no corresponding table, the contents of the pointers are zeros. The last one is a set of character classification and conversion tables which contain the same information as the single byte table except that the codes are represented as process codes and the table size is variable. The characters used for initializing values of the character classification table represent character classifications that are defined in /usr/include/ctype.h and /usr/include/wctype.h. _E1 through _E8 are for international use and _E9 through _E24 are for language dependent use.

The second output file (a data file) contains the same information, but is structured for efficient use by the character classification and conversion routines [see ctype(3C) and wctype(3W)]. The name of this output file is the value of the

character classification LC_CTYPE read in from *file*. This output file must be copied to the /usr/lib/locale/*locale*/LC_CTYPE file by someone who is super-user or a member of group bin. This file must be readable by user, group, and other; no other permissions should be set. To use the character classification and conversion tables on this file, set the LC_CTYPE category of setlocale() [see setlocale(3C)] appropriately.

The third output file (a data file) is created only if numeric editing information is specified in the input file. The name of the file is the value of the character classification LC_NUMERIC read from the *file*. This output file must be copied to the /usr/lib/locale/*locale*/LC_NUMERIC file by someone who is super-user or a member of group bin. This file must be readable by user, group, and other; no other permissions should be set. To use the numeric editing information on this file, set the LC_NUMERIC category of setlocale() appropriately.

If no input file is given, or if the argument - is encountered, wchrtbl reads from standard input.

The syntax of *file* allows the user to define the name of the data file created by wchrtbl, the assignment of characters to character classifications, the relationship between conversion letters, and byte and screen widths for up to three supplementary code sets. The keywords recognized by wchrtbl are:

LC_CTYPE	name of the first data file to be created by wchrtbl
isupper	character codes to be classified as upper-case letters
islower	character codes to be classified as lower-case letters
isdigit	character codes to be classified as numeric
isspace	character codes to be classified as spacing (delimiter) characters
ispunct	character codes to be classified as punctuation characters
iscntrl	character codes to be classified as control characters
isblank	character code for the space character
isxdigit	character codes to be classified as hexadecimal digits
ul	relationship between conversion characters
cswidth	byte and screen width information
LC_NUMERIC	name of the second data file created by wchrtbl
decimal_point	decimal delimiters
thousands_sep	thousands delimiters
grouping	A string in which each element is taken as an integer that indicates the number of digits that comprise the current group in a formatted non-monetary quantity.
LC_CTYPE1	specify that functions for specification of supplementary code set 1 follows
LC_CTYPE2	specify that functions for specification of supplementary code set 2 follows
LC_CTYPE3	specify that functions for specification of supplementary code set 3 follows

`isphonogram(iswchar1)`
: character codes to be classified as phonograms in supplementary code sets

`isideogram(iswchar2)`
: character codes to be classified as ideograms in supplementary code sets

`isenglish(iswchar3)`
: character codes to be classified as English letters in supplementary code sets

`isnumber(iswchar4)`
: character codes to be classified as numeric in supplementary code sets

`isspecial(iswchar5)`
: character codes to be classified as special letters in supplementary code sets

`iswchar6`
: character codes to be classified as other printable letters in supplementary code sets

`iswchar7 - iswchar8`
: reserved for international use

`iswchar9 - iswchar24`
: character codes to be classified as language dependent letters/characters

The keywords `iswchar1` through `iswchar24` correspond to bit names `_E1` through `_E24` defined in `wctype.h`

Any lines with the number sign (#) in the first column are treated as comments and are ignored. Blank lines are also ignored.

Characters for `isupper`, `islower`, `isdigit`, `isspace`, `ispunct`, `iscntl`, `isblank`, `isxdigit`, `ul`, `isphonogram`, `isideogram`, `isenglish`, `isnumber`, `isspecial` and `iswchar1-iswchar24` can be represented as hexadecimal or octal constants (for example, the letter `a` can be represented as `0x61` in hexadecimal or `0141` in octal) and must be up to two byte process codes. Hexadecimal and octal constants may be separated by one or more space and tab characters.

The following is the format of an input specification for `cswidth` (byte widths for supplementary code sets 2 and 3 are exclusive of the Single Shift characters):

```
cswidth n1[[:s1][,n2[:s2][,n3[:s3]]]]
```

where,

`n1` byte width for supplementary code set 1
`s1` screen width for supplementary code set 1
`n2` byte width for supplementary code set 2
`s2` screen width for supplementary code set 2
`n3` byte width for supplementary code set 3
`s3` screen width for supplementary code set 3

The dash character (-) may be used to indicate a range of consecutive numbers (inclusive of the characters delimiting the range). Zero or more space characters may be used for separating the dash character from the numbers.

The backslash character (\) is used for line continuation. Only a carriage return is permitted after the backslash character.

The relationship between conversion letters (`ul`) is expressed as ordered pairs of octal or hexadecimal constants: *<converting-character converted-character>*. These

two constants must be up to two byte process codes and may be separated by one or more space characters. Zero or more space characters may be used for separating the angle brackets (< >) from the numbers.

EXAMPLE

The following is an example of an input file used to create the JAPAN code set definition table on a file named `LC_CTYPE` and `LC_NUMERIC`.

```
#
# locale JAPAN
#
LC_CTYPE        LC_CTYPE
#
# specification for single byte characters
#
isupper   0x41 - 0x5a
islower   0x61 - 0x7a
isdigit   0x30 - 0x39
isspace   0x20   0x9 - 0xd
ispunct   0x21 - 0x2f  0x3a - 0x40  \
                0x5b - 0x60  0x7b - 0x7e
iscntrl   0x0 - 0x1f   0x7f - 0x9f
isblank   0x20
isxdigit  0x30 - 0x39 0x61 - 0x66    0x41 - 0x46
ul              <0x41 0x61> <0x42 0x62> <0x43 0x63> \
                <0x44 0x64> <0x45 0x65> <0x46 0x66> \
                <0x47 0x67> <0x48 0x68> <0x49 0x69> \
                <0x4a 0x6a> <0x4b 0x6b> <0x4c 0x6c> \
                <0x4d 0x6d> <0x4e 0x6e> <0x4f 0x6f> \
                <0x50 0x70> <0x51 0x71> <0x52 0x72> \
                <0x53 0x73> <0x54 0x74> <0x55 0x75> \
                <0x56 0x76> <0x57 0x77> <0x58 0x78> \
                <0x59 0x79> <0x5a 0x7a>
cswidth         2:2,1:1,2:2
LC_NUMERIC      LC_NUMERIC
decimal_point   .
thousands_sep
#
# specification for supplementary code set 1
#
LC_CTYPE1
isupper         0xa3c1 - 0xa3da
islower         0xa3e1 - 0xa3fa
isdigit         0xa3b0 - 0xa3b9
isspace         0xa1a1
isphonogram     0xa4a1 - 0xa4f3 0xa5a1 - 0xa5f6
isideogram      0xb0a1 - 0xb0fe 0xb1a1 - 0xb1fe 0xb2a1 - 0xb2fe \
                0xb3a1 - 0xb3fe 0xb4a1 - 0xb4fe 0xb5a1 - 0xb5fe \
                0xb6a1 - 0xb6fe 0xb7a1 - 0xb7fe 0xb8a1 - 0xb8fe \
                0xb9a1 - 0xb9fe 0xbaa1 - 0xbafe 0xbba1 - 0xbbfe \
                0xbca1 - 0xbcfe 0xbda1 - 0xbdfe 0xbea1 - 0xbefe \
                0xbfa1 - 0xbffe 0xc0a1 - 0xc0fe 0xc1a1 - 0xc1fe \
                0xc2a1 - 0xc2fe 0xc3a1 - 0xc3fe 0xc4a1 - 0xc4fe \
```

```
                0xc5a1 - 0xc5fe 0xc6a1 - 0xc6fe 0xc7a1 - 0xc7fe \
                0xcca1 - 0xccfe 0xcda1 - 0xcdfe 0xcea1 - 0xcefe \
                0xcfa1 - 0xcffe 0xd0a1 - 0xd0fe 0xd1a1 - 0xd1fe \
                0xd2a1 - 0xd2fe 0xd3a1 - 0xd3fe 0xd4a1 - 0xd4fe \
                0xd5a1 - 0xd5fe 0xd6a1 - 0xd6fe 0xd7a1 - 0xd7fe \
                0xd8a1 - 0xd8fe 0xd9a1 - 0xd9fe 0xdaa1 - 0xdafe \
                0xdba1 - 0xdbfe 0xdca1 - 0xdcfe 0xdda1 - 0xddfe \
                0xdea1 - 0xdefe 0xdfa1 - 0xdffe 0xe0a1 - 0xe0fe \
                0xe1a1 - 0xe1fe 0xe2a1 - 0xe2fe 0xe3a1 - 0xe3fe \
                0xe4a1 - 0xe4fe 0xe5a1 - 0xe5fe 0xe6a1 - 0xe6fe \
                0xe7a1 - 0xe7fe 0xe8a1 - 0xe8fe 0xe9a1 - 0xe9fe \
                0xeaa1 - 0xeafe 0xeba1 - 0xebfe 0xeca1 - 0xecfe \
                0xeda1 - 0xedfe 0xeea1 - 0xeefe 0xefa1 - 0xeffe \
                0xf0a1 - 0xf0fe 0xf1a1 - 0xf1fe 0xf2a1 - 0xf2fe \
                0xf3a1 - 0xf3fe 0xf4a1 - 0xf4fe 0xf5a1 - 0xf5fe \
                0xf6a1 - 0xf6fe 0xf7a1 - 0xf7fe 0xf8a1 - 0xf8fe \
                0xf9a1 - 0xf9fe 0xfaa1 - 0xfafe 0xfba1 - 0xfbfe \
                0xfca1 - 0xfcfe 0xfda1 - 0xfdfe 0xfea1 - 0xfefe \
isenglish       0xa3c1 - 0xa3da 0xa3e1 - 0xa3fa
isnumber        0xa3b0 - 0xa3b9
isspecial       0xa1a2 - 0xa1fe 0xa2a1 - 0xa2ae 0xa2ba - 0xa2c1 \
                0xa2ca - 0xa2d0 0xa2dc - 0xa2ea 0xa2f2 - 0xa2f9 \
                0xa2fe
iswchar6        0xa6a1 - 0xa6b8 0xa6c1 - 0xa6d8 0xa7a1 - 0xa7c1 \
                0xa7d1 - 0xa7f1
#
#               JIS X0208 whole code set
#
iswchar9        0xa1a1 - 0xa1fe 0xa2a1 - 0xa2fe 0xa3a1 - 0xa3fe \
                0xa4a1 - 0xa4fe 0xa5a1 - 0xa5fe 0xa6a1 - 0xa6fe \
                0xa7a1 - 0xa7fe 0xa8a1 - 0xa8fe 0xa9a1 - 0xa9fe \
                0xaaa1 - 0xaafe 0xaba1 - 0xabfe 0xaca1 - 0xacfe \
                0xada1 - 0xadfe 0xaea1 - 0xaefe 0xafa1 - 0xaffe \
                0xb0a1 - 0xb0fe 0xb1a1 - 0xb1fe 0xb2a1 - 0xb2fe \
                0xb3a1 - 0xb3fe 0xb4a1 - 0xb4fe 0xb5a1 - 0xb5fe \
                0xb6a1 - 0xb6fe 0xb7a1 - 0xb7fe 0xb8a1 - 0xb8fe \
                0xb9a1 - 0xb9fe 0xbaa1 - 0xbafe 0xbba1 - 0xbbfe \
                0xbca1 - 0xbcfe 0xbda1 - 0xbdfe 0xbea1 - 0xbefe \
                0xbfa1 - 0xbffe 0xc0a1 - 0xc0fe 0xc1a1 - 0xc1fe \
                0xc2a1 - 0xc2fe 0xc3a1 - 0xc3fe 0xc4a1 - 0xc4fe \
                0xc5a1 - 0xc5fe 0xc6a1 - 0xc6fe 0xc7a1 - 0xc7fe \
                0xc8a1 - 0xc8fe 0xc9a1 - 0xc9fe 0xcaa1 - 0xcafe \
                0xcba1 - 0xcbfe 0xcca1 - 0xccfe 0xcda1 - 0xcdfe \
                0xcea1 - 0xcefe 0xcfa1 - 0xcffe 0xd0a1 - 0xd0fe \
                0xd1a1 - 0xd1fe 0xd2a1 - 0xd2fe 0xd3a1 - 0xd3fe \
                0xd4a1 - 0xd4fe 0xd5a1 - 0xd5fe 0xd6a1 - 0xd6fe \
                0xd7a1 - 0xd7fe 0xd8a1 - 0xd8fe 0xd9a1 - 0xd9fe \
                0xdaa1 - 0xdafe 0xdba1 - 0xdbfe 0xdca1 - 0xdcfe \
                0xdda1 - 0xddfe 0xdea1 - 0xdefe 0xdfa1 - 0xdffe \
                0xe0a1 - 0xe0fe 0xe1a1 - 0xe1fe 0xe2a1 - 0xe2fe \
                0xe3a1 - 0xe3fe 0xe4a1 - 0xe4fe 0xe5a1 - 0xe5fe \
                0xe6a1 - 0xe6fe 0xe7a1 - 0xe7fe 0xe8a1 - 0xe8fe \
                0xe9a1 - 0xe9fe 0xeaa1 - 0xeafe 0xeba1 - 0xebfe \
```

```
                0xeca1 - 0xecfe 0xeda1 - 0xedfe 0xeea1 - 0xeefe \
                0xefa1 - 0xeffe 0xf0a1 - 0xf0fe 0xf1a1 - 0xf1fe \
                0xf2a1 - 0xf2fe 0xf3a1 - 0xf3fe 0xf4a1 - 0xf4fe \
                0xf5a1 - 0xf5fe 0xf6a1 - 0xf6fe 0xf7a1 - 0xf7fe \
                0xf8a1 - 0xf8fe 0xf9a1 - 0xf9fe 0xfaa1 - 0xfafe \
                0xfba1 - 0xfbfe 0xfca1 - 0xfcfe 0xfda1 - 0xfdfe \
                0xfea1 - 0xfefe
#
#               JIS X0208 parentheses
#
iswchar10       0xa1c6 - 0xa1db
#
#               JIS X0208 hiragana
#
iswchar11       0xa4a1 - 0xa4f3
#
#               JIS X0208 katakana
#
iswchar12       0xa5a1 - 0xa5f6
#
#               JIS X0208 other characters
#
iswchar13       0xa6a1 - 0xa6b8 0xa6c1 - 0xa6d8 0xa7a1 - 0xa7c1 \
                0xa7d1 - 0xa7f1 0xa8a1 - 0xa8bf
#
#               English letter translation table
#
ul              <0xa3c1 0xa3e1> <0xa3c2 0xa3e2> <0xa3c3 0xa3e3> \
                <0xa3c4 0xa3e4> <0xa3c5 0xa3e5> <0xa3c6 0xa3e6> \
                <0xa3c7 0xa3e7> <0xa3c8 0xa3e8> <0xa3c9 0xa3e9> \
                <0xa3ca 0xa3ea> <0xa3cb 0xa3eb> <0xa3cc 0xa3ec> \
                <0xa3cd 0xa3ed> <0xa3ce 0xa3ee> <0xa3cf 0xa3ef> \
                <0xa3d0 0xa3f0> <0xa3d1 0xa3f1> <0xa3d2 0xa3f2> \
                <0xa3d3 0xa3f3> <0xa3d4 0xa3f4> <0xa3d5 0xa3f5> \
                <0xa3d6 0xa3f6> <0xa3d7 0xa3f7> <0xa3d8 0xa3f8> \
                <0xa3d9 0xa3f9> <0xa3da 0xa3fa> \
#
#               kana translation table
#
                <0xa4a1 0xa5a1> <0xa4a2 0xa5a2> <0xa4a3 0xa5a3> \
                <0xa4a4 0xa5a4> <0xa4a5 0xa5a5> <0xa4a6 0xa5a6> \
                <0xa4a7 0xa5a7> <0xa4a8 0xa5a8> <0xa4a9 0xa5a9> \
                <0xa4aa 0xa5aa> <0xa4ab 0xa5ab> <0xa4ac 0xa5ac> \
                <0xa4ad 0xa5ad> <0xa4ae 0xa5ae> <0xa4af 0xa5af> \
                <0xa4b0 0xa5b0> <0xa4b1 0xa5b1> <0xa4b2 0xa5b2> \
                <0xa4b3 0xa5b3> <0xa4b4 0xa5b4> <0xa4b5 0xa5b5> \
                <0xa4b6 0xa5b6> <0xa4b7 0xa5b7> <0xa4b8 0xa5b8> \
                <0xa4b9 0xa5b9> <0xa4ba 0xa5ba> <0xa4bb 0xa5bb> \
                <0xa4bc 0xa5bc> <0xa4bd 0xa5bd> <0xa4be 0xa5be> \
                <0xa4bf 0xa5bf> <0xa4c0 0xa5c0> <0xa4c1 0xa5c1> \
                <0xa4c2 0xa5c2> <0xa4c3 0xa5c3> <0xa4c4 0xa5c4> \
                <0xa4c5 0xa5c5> <0xa4c6 0xa5c6> <0xa4c7 0xa5c7> \
                <0xa4c8 0xa5c8> <0xa4c9 0xa5c9> <0xa4ca 0xa5ca> \
```

```
                    <0xa4cb 0xa5cb> <0xa4cc 0xa5cc> <0xa4cd 0xa5cd> \
                    <0xa4ce 0xa5ce> <0xa4cf 0xa5cf> <0xa4d0 0xa5d0> \
                    <0xa4d1 0xa5d1> <0xa4d2 0xa5d2> <0xa4d3 0xa5d3> \
                    <0xa4d4 0xa5d4> <0xa4d5 0xa5d5> <0xa4d6 0xa5d6> \
                    <0xa4d7 0xa5d7> <0xa4d8 0xa5d8> <0xa4d9 0xa5d9> \
                    <0xa4da 0xa5da> <0xa4db 0xa5db> <0xa4dc 0xa5dc> \
                    <0xa4dd 0xa5dd> <0xa4de 0xa5de> <0xa4df 0xa5df> \
                    <0xa4e0 0xa5e0> <0xa4e1 0xa5e1> <0xa4e2 0xa5e2> \
                    <0xa4e3 0xa5e3> <0xa4e4 0xa5e4> <0xa4e5 0xa5e5> \
                    <0xa4e6 0xa5e6> <0xa4e7 0xa5e7> <0xa4e8 0xa5e8> \
                    <0xa4e9 0xa5e9> <0xa4ea 0xa5ea> <0xa4eb 0xa5eb> \
                    <0xa4ec 0xa5ec> <0xa4ed 0xa5ed> <0xa4ee 0xa5ee> \
                    <0xa4ef 0xa5ef> <0xa4f0 0xa5f0> <0xa4f1 0xa5f1> \
                    <0xa4f2 0xa5f2> <0xa4f3 0xa5f3>
        #
        # specification for supplementary code set 2
        #
        LC_CTYPE2
        iswchar6        0xa1 - 0xdf
        iswchar14       0xa1 - 0xdf
```

FILES

`/usr/lib/locale/`*locale*`/LC_CTYPE`
: data files containing character classification and conversion tables and character set width information created by `chrtbl` or `wchrtbl`.

`/usr/lib/locale/`*locale*`/LC_NUMERIC`
: data files containing numeric editing information.

`/usr/include/ctype.h`
: header file containing information used by character classification and conversion routines for single byte characters.

`/usr/include/wctype.h`
: header file containing information used by international character classification and conversion routines for supplementary code sets.

`/usr/include/xctype.h`
: header file containing information used by language dependent character classification and conversion routines for supplementary code sets.

DIAGNOSTICS

The error messages produced by `wchrtbl` are intended to be self-explanatory. They indicate errors in the command line or syntactic errors encountered within the input file.

SEE ALSO

`ctype`(3C), `setlocale`(3C), `environ`(5).

NOTE

The `numeric` entry is used to specify decimal and thousands delimiters by `wchrtbl` of the previous release of *MNLS*. In *SVR4 MNLS*, the `decimal_point` and `thousands_sep` entries are used instead of the `numeric`, to adopt its syntax with that of `chrtbl`(1).

NAME

whodo – who is doing what

SYNOPSIS

/usr/sbin/whodo

DESCRIPTION

Whodo produces formatted and dated output from information in the /etc/utmp and /etc/ps_data files.

The display is headed by the date, time, and machine name. For each user logged in, device name, user-id, and login time are shown, followed by a list of active processes associated with the user-id. The list includes the device name, process-id, cpu minutes and seconds used, and process name.

EXAMPLE

The command:

```
whodo
```

produces a display like this:

```
Tue Mar 12 15:48:03 1985
bailey

tty09    mcn        8:51
    tty09   28158    0:29 sh

tty52    bdr       15:23
    tty52   21688    0:05 sh
    tty52   22788    0:01 whodo
    tty52   22017    0:03 vi
    tty52   22549    0:01 sh
```

FILES

/etc/passwd
/etc/ps_data
/etc/utmp

SEE ALSO

ps(1), who(1) in the *User's Reference for the DG/UX System*.

NAME

wmtd – start the WORM magnetic tape device server

SYNOPSIS

/usr/sbin/wmtd *wdevice=pdevice* ...

where:

wdevice Device number in the /dev/wmt directory
pdevice Pathname of the physical device

DESCRIPTION

A WORM drive is a write-once read-many disk device. The WORM as magnetic tape server (daemon), wmtd, is designed to make, as much as possible, a WORM disk device act like a magnetic tape device. From the user's perspective, all of the system tape archiving commands, such as mt(1), dump2(1M), restore(1M), and sysadm(1M) will behave as they do when archiving to magnetic tapes. An exception to this rule is that a WORM disk may be written only once, a feature that makes WORM drives a good choice for permanent archives.

A user opens a special file in the directory /dev/wmt. The DG/UX kernel then communicates with the wmtd process to perform operations on the WORM device. The wmtd process knows the physical device with which to communicate by the logical-to-physical device mappings specified on the command-line. For example, if the system administrator wants /dev/wmt/0 and /dev/wmt/0n to be associated with the device, /dev/rpdsk/2, then the mapping would be 0=/dev/rpdsk/2. More than one device mapping may be specified when the system has more than one WORM device, but only one device may be accessed at a time.

The wmtd server automatically puts itself in the background and detaches from any controlling terminal. Unanticipated errors are communicated to the system through syslogd(1M). Only a superuser can start the wmtd program.

The preferred way to start the server is to let the system start it at boot time. To have the system start wmtd, the system administrator must modify the /etc/dgux.params initialization file. The variable wmtd_START should be set to "true" and the wmtd_ARG should contain the command line arguments.

The protocol used between wmt(7) and wmtd is the same protocol used by pmtd(1M).

FILES

/usr/include/sys/errno.h File describing DG/UX errno values.

SEE ALSO

pmtd(1M), mt(1), dump2(1M), restore(1M), sysadm(1M), cpio(1), kill(2), ioctl(2), wmt(7), syslogd(1M).

CAVEATS

The ioctl(2) operations (with the command MTIOCTOP) supported by the wmtd server are as follows: MTFSF, MTBSF, MTREW, MTOFFL, MTWEOF, and MTNOP.

If a /dev/wmt device is specified as the input-output device using one of the archiving commands and no valid mapping exists, wmtd returns ENODEV (in errno) to the calling process.

The superuser should never send a SIGKILL (i.e. kill –9) signal to wmtd. A SIGTERM (i.e. kill with no options) signal will allow the server to "clean up" any read or write that may be in progress. If the server is sent a SIGKILL signal when writing, the remainder of the WORM cartridge will most likely be unusable.

When using `cpio`(1), the superuser should specify the `-B` switch, as the `wmtd` server is slow when using small buffers. For maximum efficiency, the buffer size should be a multiple of the WORM device's sector size.

NAME

xdrtoc – convert distribution table of contents into ascii format

SYNOPSIS

xdrtoc [*file*]

DESCRIPTION

Xdrtoc translates the XDR-format table of contents found on software distribution tapes into a human readable form. This information is written to the standard output.

Arguments

If no arguments are given, then xdrtoc reads the XDR table of contents from its standard input. Otherwise, it reads from *file*.

FILES

None

SEE ALSO

xdr(3N).

NAME

ypinit - build and install Network Information Service database

SYNOPSIS

/usr/etc/yp/ypinit -m

/usr/etc/yp/ypinit -s *master_name*

DESCRIPTION

ypinit sets up a Network Information Service database on an NIS server. It can be used to set up a master or a slave server. You must be the super-user to run it. It asks a few, self-explanatory questions, and reports success or failure to the terminal.

It sets up a master server using the simple model in which that server is master to all maps in the data base. This is the way to bootstrap the NIS system; later if you want you can change the association of maps to masters. All databases are built from scratch, either from information available to the program at runtime, or from the ASCII data base files in /etc. Some of these files are listed below under FILES. All such files should be in their "traditional" form, rather than the abbreviated form used on client machines.

An NIS database on a slave server is set up by copying an existing database from a running server. The *master_name* argument should be the hostname of NIS server (either the master server for all the maps, or a server on which the data base is up-to-date and stable).

Read ypfiles(4) and ypserv(1M) for an overview of the Network Information Service.

OPTIONS

-m Indicate that the local host is to be the NIS master.

-s Set up a slave database.

FILES

/etc/passwd
/etc/group
/etc/hosts
/etc/networks
/etc/services
/etc/protocols
/etc/ethers

SEE ALSO

makedbm(1M), ypmake(1M), yppush(1M), ypserv(1M), ypxfr(1M), ypfiles(4).

NAME

ypmake – rebuild Network Information Service database

SYNOPSIS

cd /etc/yp; make [*map*]

DESCRIPTION

The file called Makefile in /etc/yp is used by make to build the Network Information Service database. With no arguments, make creates dbm databases for any NIS maps that are out-of-date, and then executes yppush(1M) to notify slave databases that there has been a change.

If you supply a *map* on the command line, make will update that map only. Typing make passwd will create and yppush the password database (assuming it is out of date). Likewise, make hosts and make networks will create and yppush the host and network files, /etc/hosts and /etc/networks.

There are four special variables used by make: SRC_DIR, which gives the directory of the source files; NOPUSH, which when non-null inhibits doing a yppush of the new database files; DOM, used to construct a domain other than the master's default domain and INTERDOMAIN, which, when set to -b, allows NIS to access the domain name server. The domain name server must be set up and configured for the INTERDOMAIN variable to work (see Chapter 3, "The Network Information Service," in *Managing ONC™/NFS® and Its Facilities on the DG/UX System* for more information). The default for SRC_DIR is /etc, and the default for NOPUSH is the null string.

Refer to ypfiles(4) and ypserv(1M) for an overview of the NIS.

FILES

/etc/yp
/etc/hosts
/etc/networks

SEE ALSO

make(1), makedbm(1M), yppush(1M), ypserv(1M), ypfiles(4).

NAME

yppasswdd - server for modifying Network Information Service password file

SYNOPSIS

/usr/etc/rpc.yppasswdd *filename* [*adjunct_file*] [-m *argument1 argument2 ...*]

DESCRIPTION

Yppasswdd is a server that handles password change requests from yppasswd(1). It changes a password entry in *filename*, which is assumed to be in the format of passwd(4).

If the -m option is given, then after *filename* is modified, a make(1) will be performed in /etc/yp. Any arguments following the flag will be passed to make.

This server is not run by default, nor can it be started up from inetd(1M). If it is desired to enable remote password updating for the Network Information Service, then an entry for yppasswdd should be put in the /etc/nfs.params file of the host serving as the master for the Network Information Service passwd file.

EXAMPLE

If the Network Information Service password file is stored as /etc/yp/src/passwd, then to have password changes propagated immediately, the server should be invoked as

```
/usr/etc/rpc.yppasswdd /etc/yp/src/passwd -m SRC_DIR=/etc/yp/src passwd
```

In this case, src is the NIS domain name.

FILES

/etc/yp/Makefile
/etc/nfs.params

SEE ALSO

make(1), yppasswd(1), inetd(1M), ypmake(1M), passwd(4), ypfiles(4).

NAME

yppoll - what version of an NIS map is at an NIS server host

SYNOPSIS

/usr/etc/yp/yppoll [-h *host*] [-d *domain*] *mapname*

DESCRIPTION

Yppoll asks a ypserv(1M) process what the order number is, and which host is the master NIS server for the named map. If the server is a v.1 NIS protocol server, yppoll uses the older protocol to communicate with it. In this case, it also uses the older diagnostic messages in case of failure.

OPTIONS

-h *host* Ask the ypserv process at *host* about the map parameters. If *host* is not specified, the NIS server for the local host is used. That is, the default host is the one returned by ypwhich(1M).

-d *domain*
Use *domain* instead of the default domain.

SEE ALSO

ypserv(1M), ypwhich(1M), ypfiles(4).

NAME

yppush - force propagation of a changed NIS map

SYNOPSIS

/usr/etc/yp/yppush [-v] [-d *domain*] *mapname*

DESCRIPTION

Yppush copies a new version of a Network Information Service (NIS) map from the master NIS server to the slave NIS servers. It is normally run only on the master NIS server by the Makefile in /etc/yp after the master databases are changed. It first constructs a list of NIS server hosts by reading the NIS map ypservers within the *domain*. Keys within the map ypservers are the ASCII names of the machines on which the NIS servers run.

A "transfer map" request is sent to the NIS server at each host, along with the information needed by the transfer agent (the program which actually moves the map) to call back the yppush. When the attempt has completed (successfully or not), and the transfer agent has sent yppush a status message, the results may be printed to stdout. Messages are also printed when a transfer is not possible; for instance when the request message is undeliverable, or when the timeout period on responses has expired.

Refer to ypfiles(4) and ypserv(1M) for an overview of the Network Information Service.

OPTIONS

-d Specify a *domain* instead of the default domain.

-v Verbose. This causes messages to be printed when each server is called, and for each response. If this flag is omitted, only error messages are printed.

FILES

/etc/yp/*domain*/ypservers.{dir, *pag*}

/etc/yp

SEE ALSO

ypserv (1M), ypxfr (1M), ypfiles (4), NIS protocol specification.

BUGS

In the current implementation (version 2 NIS protocol), the transfer agent is ypxfr(1M), which is started by the ypserv program. If yppush detects that it is speaking to a version 1 NIS protocol server, it uses the older protocol, sending a version 1 YPPROC_GET request and issues a message to that effect. Unfortunately, there is no way of knowing if or when the map transfer is performed for version 1 servers. yppush prints a message saying that an "old-style" message has been sent. The system administrator should later check to see that the transfer has actually taken place.

NAME

ypserv, ypbind – Network Information Service server and binder processes

SYNOPSIS

/usr/etc/ypserv

/usr/etc/ypbind

DESCRIPTION

The Network Information Service (NIS) provides a simple network lookup service consisting of databases and processes. The databases are dbm(3X) files in a directory tree rooted at /etc/yp. These files are described in ypfiles(4). The processes are /usr/etc/ypserv, the NIS database lookup server, and /usr/etc/ypbind, the NIS binder. The programmatic interface to NIS is described in ypclnt(3N). Administrative tools are described in yppush(1M), ypxfr(1M), yppoll(1M), ypwhich(1M), and ypset(1M). Tools to see the contents of NIS maps are described in ypcat(1), and ypmatch(1). Database generation and maintenance tools are described in ypinit(1M), ypmake(1M), and makedbm(1M).

Both ypserv and ypbind are server (daemon) processes typically activated at system startup time from /usr/sbin/init.d/rc.ypserv. ypserv runs only on NIS server machines with a complete NIS database. ypbind runs on all machines using NIS services, both NIS servers and clients.

The primary function of ypserv is to look up information in its local database of NIS maps. The operations performed by ypserv are defined for the implementor by the *NIS Protocol Specification*, and for the programmer by the header file <rpcsvc/yp_prot.h>. Communication to and from ypserv is by means of RPC calls. Lookup functions are described in ypclnt(3N), and are supplied as C-callable functions in the C library. There are four lookup functions, all of which are performed on a specified map within some NIS domain: *Match*, *Get_first*, *Get_next*, and *Get_all*. The *Match* operation takes a key, and returns the associated value. The *Get_first* operation returns the first key-value pair from the map, and *Get_next* can be used to enumerate the remainder. *Get_all* ships the entire map to the requester as the response to a single RPC request.

Two other functions supply information about the map, rather than map entries: *Get_order_number*, and *Get_master_name*. In fact, both order number and master name exist in the map as key-value pairs, but the server will not return either through the normal lookup functions. (If you examine the map with makedbm(1M), however, they will be visible.) Other functions are used within the NIS subsystem itself, and are not of general interest to NIS clients. They include *Do_you_serve_this_domain?*, *Transfer_map*, and *Reinitialize_internal_state*.

The function of ypbind is to remember information that lets client processes on a single node communicate with some ypserv process. ypbind must run on every machine which has NIS client processes; ypserv may or may not be running on the same node, but must be running somewhere on the network.

The information ypbind remembers is called a *binding* — the association of a domain name with the internet address of the NIS server, and the port on that host at which the ypserv process is listening for service requests. This information is cached in the directory /etc/yp/binding using a filename of domainname.*version*.

The process of binding is driven by client requests. As a request for an unbound domain comes in, the ypbind process broadcasts on the net trying to find a ypserv process that serves maps within that domain. Since the binding is established by

broadcasting, there must be at least one ypserv process on every net. Once a domain is bound by a particular ypbind, that same binding is given to every client process on the node. The ypbind process on the local node or a remote node may be queried for the binding of a particular domain by using the ypwhich(1) command.

Bindings and rebindings are handled transparently by the C library routines. If ypbind is unable to speak to the ypserv process it's bound to, it marks the domain as unbound, tells the client process that the domain is unbound, and tries to bind the domain once again. Requests received for an unbound domain will wait until the domain requested is bound. In general, a bound domain is marked as unbound when the node running ypserv crashes or gets overloaded. In such a case, ypbind will try to bind to any NIS server (typically one that is less-heavily loaded) available on the net.

ypbind also accepts requests to set its binding for a particular domain. The request is usually generated by the NIS subsystem itself. ypset(1M) is a command to access the *Set_domain* facility. It is for unsnarling messes. Note that the *Set_domain* procedure only accepts requests from processes running as root.

FILES

If the file /etc/yp/ypserv.log exists when ypserv starts up, log information will be written to this file when error conditions arise.

The file(s) /etc/yp/binding/domainname.version will be created to speed up the binding process. These files cache the last successful binding created for the given domain, when a binding is requested these files are checked for validity and then used.

/etc/yp
/usr/etc/ypbind

SEE ALSO

domainname(1), ypcat(1), ypmatch(1), makedbm(1M), ypmake(1M), ypinit(1M), yppoll(1M), yppush(1M), ypset(1M), ypwhich(1), ypxfr(1M), dbm(3X), ypclnt(3N), ypfiles(4).

NOTES

Both ypbind and ypserv support multiple domains. The ypserv process determines the domains it serves by looking for directories of the same name in the directory /etc/yp. It will reply to all broadcasts requesting yp service for that domain. Additionally, the ypbind process can maintain bindings to several domains and their servers, the default domain is however the one specified by the domainname(1) command at startup time.

NAME

ypset - point ypbind at a particular server

SYNOPSIS

/usr/etc/yp/ypset [-V1|-V2] [-d *domain*] [-h *host*] *server*

DESCRIPTION

ypset tells ypbind to get NIS services for the specified *domain* from the ypserv process running on *server*. If *server* is down, or isn't running ypserv, this is not discovered until an NIS client process tries to get a binding for the domain. At this point, the binding set by ypset will be tested by ypbind. If the binding is invalid, ypbind will attempt to rebind for the same domain. You must be root to execute this command.

ypset is useful for binding a client node which is not on a broadcast net, or is on a broadcast net which isn't running an NIS server host. It also is useful for debugging NIS client applications, for instance where an NIS map only exists at a single NIS server host.

In cases where several hosts on the local net are supplying NIS services, it is possible for ypbind to rebind to another host even while you attempt to find out if the ypset operation succeeded. For example, you can type:

```
example% ypset host1
example% ypwhich
host2
```

which can be confusing. This is a function of the NIS subsystem's attempt to load-balance among the available NIS servers, and occurs when *host1* does not respond to ypbind because it is not running ypserv (or is overloaded), and *host2*, running ypserv, gets the binding.

server indicates the NIS server to bind to, and can be specified as a name or an IP address. If specified as a name, ypset will attempt to use NIS services to resolve the name to an IP address. This will work only if the node has a current valid binding for the domain in question. In most cases, *server* should be specified as an IP address.

Refer to ypfiles(4) and ypserv(1M) for an overview of the Network Information Service.

OPTIONS

-V1 Bind *server* for the (old) v.1 NIS protocol.

-V2 Bind *server* for the (current) v.2 NIS protocol.

If no version is supplied, ypset, first attempts to set the domain for the (current) v.2 protocol. If this attempt fails, ypset, then attempts to set the domain for the (old) v.1 protocol.

-h*host* Set ypbind's binding on *host*, instead of locally. *host* can be specified as a name or as an IP address. Note that in the DG/UX System, only requests generated locally by the root user are allowed.

-d*domain*
Use *domain* instead of the default domain.

SEE ALSO

ypwhich(1), ypserv(1M), ypfiles(4).

NAME

ypupdated – server for changing NIS information

SYNOPSIS

```
/usr/etc/rpc.ypupdated [ -is ]
```

DESCRIPTION

Ypupdated is a server (daemon) that updates information in the Network Information Service. ypupdated consults the file updaters(4) in the directory /etc/yp to determine which NIS maps should be updated and how to change them.

By default, the server requires the most secure method of authentication available to it, either DES (secure) or UNIX (insecure).

OPTIONS

-s accept only calls authenticated using the secure RPC mechanism (AUTH_DES authentication). This disables programmatic updating of NIS maps unless the network supports these calls.

-i also accept RPC calls with the insecure AUTH_UNIX credentials. This allows programmatic updating of NIS maps in all networks.

FILES

/etc/yp/updaters

SEE ALSO

inetd(1M), keyserv(1M), updaters(4).

NAME

ypxfr - transfer NIS map from an NIS server to here

SYNOPSIS

/usr/etc/yp/ypxfr [-f] [-c] [-d *domain*] [-h *host*] [-s *domain*] [-C *tid prog ipaddr port*] [-S] *mapname*

DESCRIPTION

ypxfr moves an NIS map in the default domain for the local host to the local host by making use of normal NIS services. It creates a temporary map in the directory /etc/yp/*domain* (this directory must already exist; *domain* is the default domain for the local host), fills it by enumerating the map's entries, fetches the map parameters (master and order number) and loads them. It then deletes any old versions of the map and moves the temporary map to the real *mapname*. Note that you must be root to execute this command.

If run interactively, ypxfr writes its output to the terminal. However, if it is invoked without a controlling terminal, and if the log file /etc/yp/ypxfr.log exists, it will append all its output to that file. Since ypxfr is most often run from the crontab file, or by ypserv, you can use the log file to retain a record of what was attempted and what the results were.

For consistency between servers, ypxfr should be run periodically for every map in the NIS data base. Different maps change at different rates: the *services.byname* map may not change for months at a time, for instance, and may therefore be checked only once a day (in the wee hours). You may know that *mail.aliases* or *hosts.byname* changes several times per day. In such a case, you may want to check hourly for updates. A crontab(4) entry can be used to perform periodic updates automatically. Rather than having a separate crontab entry for each map, you can group commands to update several maps in a shell script. Examples (mnemonically named) are in /usr/etc/yp: ypxfr_1perday, ypxfr_2perday, and ypxfr_1perhour. They can serve as reasonable first cuts.

Refer to ypfiles(4) and ypserv(1M) for an overview of the Network Information Service.

OPTIONS

-f Force the transfer to occur even if the version at the master is not more recent than the local version.

-c Do not send a "Clear current map" request to the local ypserv process. Use this flag if ypserv is not running locally at the time you are running ypxfr. Otherwise, ypxfr will complain that it can't talk to the local ypserv, and the transfer will fail.

-d*domain*
Specify a domain other than the default domain.

-h*host* Get the map from *host*, regardless of what the map says the master is. If *host* is not specified, ypxfr will ask the NIS service for the name of the master, and try to get the map from there. *host* may be a name or an internet address in the form *a.b.c.d*.

-s*domain*
Specify a source domain from which to transfer a map that should be the same across domains (such as the *services.byname* map).

-C*tid prog ipaddr port*
This option is only for use by ypserv. When ypserv invokes ypxfr, it

specifies that `ypxfr` should call back a `yppush` process at the host with IP address *ipaddr*, registered as program number *prog*, listening on port *port* and waiting for a response to transaction *tid*.

`-S` This option causes `ypxfr` to require that the `ypserv` server, from which it will obtain the maps to be transferred, is using "privileged" IP ports. Since only super-user processes are typically allowed to use privileged ports, this feature adds an extra measure of security to the transfer. If the map being transferred is a secure map, `ypxfr` sets the permissions on the map to 0600.

FILES

`/etc/yp/ypxfr.log`
log file

`/usr/etc/yp/ypxfr_1perday`
script to run one transfer per day, for use with `cron`(1M)

`/usr/etc/yp/ypxfr_2perday`
script to run two transfers per day

`/usr/etc/yp/ypxfr_1perhour`
script for hourly transfers of volatile maps

`/etc/yp/`*domain* NIS domain

SEE ALSO

`cron`(1M), `yppush`(1M), `ypserv`(1M), `crontab`(4), `ypfiles`(4).

NAME

zdump – time zone dumper

SYNOPSIS

zdump [-v] [-c *cutoffyear*] *zonename* ...

DESCRIPTION

The zdump command prints the current time in each *zonename* named on the command line.

The following options are available:

-v For each *zonename* on the command line, print the current time, the time at the lowest possible time value, the time one day after the lowest possible time value, the times both one second before and exactly at each time at which the rules for computing local time change, the time at the highest possible time value, and the time at one day less than the highest possible time value. Each line ends with isdst=1 if the given time is Daylight Saving Time or isdst=0 otherwise.

-c *cutoffyear*
Cut off the verbose output near the start of the year *cutoffyear*.

FILES

/usr/lib/locale/TZ
standard zone information directory

SEE ALSO

zic(1M), ctime(3C).

NAME

zic – time zone compiler

SYNOPSIS

zic [-v] [-d *directory*] [-l *timezone*] [*filename* ...]

where:

directory The pathname of a directory containing timezone information; default = /usr/lib/locale/TZ

timezone The name of a time zone to use as local time

filename The name of the file from which to take input

DESCRIPTION

Zic reads text from the file(s) named on the command line and creates the time conversion information files specified in this input. If a *filename* is '–', the standard input is read.

Input lines are made up of fields. Fields are separated by any number of white space characters. Leading and trailing white space on input lines is ignored. A pound sign (#) in the input introduces a comment which extends to the end of the line the pound sign appears on. White space characters and pound signs may be enclosed in double quotes (") if they're to be used as part of a field. Any line that is blank (after comment stripping) is ignored. Non-blank lines are expected to be of one of three types: rule lines, zone lines, and link lines.

A rule line has the form

Rule *NAME FROM TO TYPE IN ON AT SAVE LETTER/S*

For example:

```
Rule   USA   1969 1973   -   Apr lastSun 2:00 1:00     D
```

The fields that make up a rule line are:

NAME Gives the (arbitrary) name of the set of rules this rule is part of.

FROM Gives the first year in which the rule applies. The word minimum (or an abbreviation) means the minimum year with a representable time value. The word maximum (or an abbreviation) means the maximum year with a representable time value.

TO Gives the final year in which the rule applies. In addition to minimum and maximum (as above), the word only (or an abbreviation) may be used to repeat the value of the FROM field.

TYPE Gives the type of year in which the rule applies. If TYPE is '–' then the rule applies in all years between FROM and TO inclusive; if TYPE is uspres, the rule applies in U.S. Presidential election years; if TYPE is nonpres, the rule applies in years other than U.S. Presidential election years. If TYPE is something else, then zic executes the command

yearistype *year type*

to check the type of a year: an exit status of zero is taken to mean that the year is of the given type; an exit status of one is taken to mean that the year is not of the given type.

IN Names the month in which the rule takes effect. Month names may be abbreviated.

ON Gives the day on which the rule takes effect. Recognized forms include:

`5`	the fifth of the month
`lastSun`	the last Sunday in the month
`lastMon`	the last Monday in the month
`Sun>=8`	first Sunday on or after the eighth
`Sun<=25`	last Sunday on or before the 25th

Names of days of the week may be abbreviated or spelled out in full. Note: there must be no spaces within the `ON` field.

AT Gives the time of day at which the rule takes effect. Recognized forms include:

`2`	time in hours
`2:00`	time in hours and minutes
`15:00`	24-hour format time (for times after noon)
`1:28:14`	time in hours, minutes, and seconds

Any of these forms may be followed by the letter `w` if the given time is local "wall clock" time or `s` if the given time is local "standard" time; in the absence of `w` or `s`, wall clock time is assumed.

`SAVE` Gives the amount of time to be added to local standard time when the rule is in effect. This field has the same format as the `AT` field (although, of course, the `w` and `s` suffixes are not used).

`LETTER/S` Gives the "variable part" (for example, the "S" or "D" in "EST" or "EDT") of time zone abbreviations to be used when this rule is in effect. If this field is '-', the variable part is null.

A zone line has the form

Zone *NAME GMTOFF RULES/SAVE FORMAT [UNTIL]*

For example:

```
Zone   Australia/South-west  GMTOFF   RULES/SAVE   FORMAT
```

The fields that make up a zone line are:

NAME The name of the time zone. This is the name used in creating the time conversion information file for the zone.

GMTOFF The amount of time to add to GMT to get standard time in this zone. This field has the same format as the `AT` and `SAVE` fields of rule lines; begin the field with a minus sign if time must be subtracted from GMT.

RULES/SAVE
The name of the rule(s) that apply in the time zone or, alternately, an amount of time to add to local standard time. If this field is '-' then standard time always applies in the time zone.

FORMAT The format for time zone abbreviations in this time zone. The pair of characters `%s` is used to show where the "variable part" of the time zone abbreviation goes. `UNTIL` The time at which the GMT offset or the rule(s) change for a location. It is specified as a year, a month, a day, and a time of day. If this is specified, the time zone information is generated from the given GMT offset and rule change until the time specified.

The next line must be a "continuation" line; this has the same form as a zone line except that the string "Zone" and the name are omitted, as the continuation line will place information starting at the time specified as the `UNTIL` field in the previous line in the file used by the previous

line. Continuation lines may contain an UNTIL field, just as zone lines do, indicating that the next line is a further continuation.

A link line has the form

```
Link  LINK-FROM    LINK-TO
```

For example:

```
Link  US/Eastern   EST5EDT
```

The LINK-FROM field should appear as the NAME field in some zone line; the LINK-TO field is used as an alternate name for that zone.

Except for continuation lines, lines may appear in any order in the input.

Options

-v Complain if a year that appears in a data file is outside the range of years representable by system time values (0:00:00 AM GMT, January 1, 1970, to 3:14:07 AM GMT, January 19, 2038).

-d *directory*
Create time conversion information files in the directory directory rather than in the standard directory /usr/lib/locale/TZ.

-l *timezone*
Use the time zone timezone as local time. zic will act as if the file contained a link line of the form

```
Link  timezone          localtime
```

FILES

/usr/lib/locale/TZ
standard directory used for created files

SEE ALSO

time(1), ctime(3C).

NOTE

For areas with more than two types of local time, you may need to use local standard time in the AT field of the earliest transition time's rule to ensure that the earliest transition time recorded in the compiled file is correct.

End of Chapter

Chapter 2
System Calls

This chapter is a place holder to make the chapter numbers match the man page numbers. The DG/UX system call man pages are in Chapter 2 of the *Programmer's Reference for the DG/UX System (Volume 1)*.

End of Chapter

Chapter 3
Subroutines and Libraries

This chapter is a place holder to make the chapter numbers match the man page numbers. The DG/UX subroutine and library man pages are in Chapter 3 of the *Programmer's Reference for the DG/UX System (Volume 2)*.

End of Chapter

Chapter 4
File Formats

This chapter contains in printed form the online manual entries for formats of DG/UX and TCP/IP system administration files. The first entry, **intro**(4M), gives an introduction. The remaining entries are in alphabetical order.

These man pages document the structure of particular kinds of files; for example, the format of the **/etc/sysadm/dumpcyle** file used by **admbackup**(1M) is given in **dumpcycle**(4M). In general, the C language structures corresponding to these formats can be found in the directories **/usr/include** and **/usr/include/sys**.

For a description of other file formats, see Chapter 4 of the *Programmer's Reference for the DG/UX System*.

In Revision 03 of this manual, the man pages for TCP/IP administrative files were added. Table 4-1 summarizes the TCP/IP administrative files.

Table 4-1 Summary of TCP/IP Administrative Files

Name	Description
gateways(4M)	Database for **routed**
hosts.equiv(4M)	List of trusted hosts
inetd.conf(4M)	Internet services database
pmterrtab(4M)	Table of equivalent error numbers
pmttapetab(4M)	Table of remote tape devices
resolv.conf(4M)	Configuration file for domain name resolver
snmpd_files(4M)	Configuration files for **snmpd**: **snmpd.config**, **snmpd.communities**, and **snmpd.trap_communities**
tcpip.params(4M)	Network parameter database for rc scripts

In addition, the following man pages are new in Revision 03:

cpz(4M)
vtc.addrs(4M)

NAME

`intro` – introduction to file formats

DESCRIPTION

This section outlines the formats of various files. The C structure declarations for the file formats are given where applicable. Usually, the header files containing these structure declarations can be found in the directories `/usr/include` or `/usr/include/sys`. For inclusion in C language programs, however, the syntax `#include` <*filename.h*> or `#include` <`sys`/*filename.h*> should be used.

NAME

cpz - compose-key maps

SYNOPSIS

/usr/lib/kbd/*.cpz

DESCRIPTION

These files contain mapping tables to be used by the att_kbd keyboard STREAMS module (see att_kbd(7) and kbdcomp(1M)). These mapping tables allow the user to compose characters and symbols using a keyboard that does not offer keys for the desired characters or symbols. By convention, the "compose key" is ^T (control-T). In general, the compose key is followed by a two-character sequence. The entire three-character sequence is mapped to the desired code by the att_kbd module, based on the contents of the table. To transmit a control-T literally, rather than to have it be interpretted as part of a compose key sequence, type control-T control-T.

EXAMPLES

The compose key table for 8859-1 (**/usr/lib/kbd/88591.cpz**) contains the following mappings. To enter any of the characters listed here, you must type control-T followed by one of the character pair(s) listed under "Keys Pressed". Usually you can reverse the two keys you must press. For example, if "ab" is listed, "ba" generally works as well. Alternative key sequences are listed for some of the characters. "<space>" means you should press the space bar.

Character	Keys Pressed
non-break space	<space><space>
quotation mark	"<space>
number sign	++
apostrophe	'<space>
commercial at	AA
opening bracket	((
backslash	// /<
closing bracket	))
circumflex accent	^<space> ><space>
grave accent	`<space>
opening brace	(-
vertical line	/^ VL vl
closing brace	-)
tilde	~<space>
inverted !	!!
cent sign	c/ C/ c| C|
pound sign	l- L- l= L=
currency sign	xo XO xO Xo x0 X0
yen sign	y- Y- y= Y=
broken vertical bar	!^ || VB vb
section sign	So SO sO so S0 s0 S! s!
diaeresis	""
copyright	cO Co CO co c0 C0
Female Ordinal	a_ A_
left angle quotation mark	<<
logical not	-,

SHY	--
registered trademark	RO
macron	-^ —
Degree sign	0^
Degree sign	0*
Plus minus	+-
Superscript 2	2^ 2S 2s
Superscript 3	3^ 3S 3s
acute accent	”
micron sign	u/ U/
paragraph sign	p! P!
middle dot	.^ ..
cedilla	,,
Superscript 1	1^ 1s 1S
Masculine Ordinal	o_ O_
right angle quotation mark	>>
one quarter	14
one half	12
three quarters	34
inverted ?	??
A grave	A‘
A acute	A’
A circumflex	A^ A>
A tilde	A A-
A umlaut	A"
A circle	A*
AE dipthong	AE
C cedilla	C,
E grave	E‘
E acute	E’
E circumflex	E^ E>
E umlaut	E"
I grave	I‘
I acute	I’
I circumflex	I^ I>
I umlaut	I"
capital Iclandic eth	D-
N tilde	N N-
O grave	O‘
O acute	O’
O circumflex	O^ O>
O tilde	O O-
O umlaut	O"
multiply	xx
O slash	O/
U grave	U‘
U acute	U’
U circumflex	U^ U>
U umlaut	U"

Y acute	Y'
capital Iclandic thor	TH
sharp s	ss
a grave	a‘
a acute	a'
a circumflex	a^ a>
a tilde	a a-
a umlaut	a"
a circle	a*
ae dipthong	ae
c cedilla	c,
e grave	e‘
e acute	e'
e circumflex	e^ e>
e umlaut	e"
i grave	i‘
i acute	i'
i circumflex	i^ i>
i umlaut	i"
small Iclandic eth	d-
n tilde	n n-
o grave	o‘
o acute	o'
o circumflex	o^ o>
o tilde	o o-
o umlaut	o"
divides	-: -;
o slash	o/
u grave	u‘
u acute	u'
u circumflex	u^ u>
u umlaut	u"
y acute	y'
small Iclandic thor	th
y umlaut	y"

FILES

`/usr/lib/kbd/*.cpz` — Compiled mapping tables.

SEE ALSO

`att_kbd`(7), `kbdcomp`(1M), `kbdset`(1), `kbdload`(1M)

NAME

dfm - DOS file manager

DESCRIPTION

The DG/UX kernel provides configurable support for PC DOS formatted floppies in 4 different formats. There is support for the high and low density versions of the 5.25" and 3.5" floppy disk drives. A high density 5.25" floppy holds 1.2 megabytes while a low density 5.25" floppy holds 360kb. A high density 3.5" floppy holds 1.4 megabytes while a low density one holds 720kb.

The DOS file manager allows the system administrator to mount a DOS floppy into the UNIX file system hierarchy. A mounted DOS floppy will appear as a UNIX file system with some restrictions imposed by the DOS file system structure. There are only two basic file types supported in this file system, ordinary files and directories. Hidden and system files will be displayed, but cannot be created. The mode of all files from the DOS file system will be read/write and executable for user, group and other. This will be true even if the floppy is mounted readonly or rendered readonly in a physical manner. However, you will not be able to modify such a floppy. Attempting to will result in an error.

In DOS, there is a restriction on the names of files and directories. DOS filenames come from a more restrictive character set than normal DG/UX filenames. First, there is no case sensitivity in DOS filenames. The DOS file manager will translate all input filenames to upper case for storage on the floppy, and display all filename characters found on the floppy as lower case when outputing them to the user. Secondly, names on the DOS file system are restricted to two naming components, a base component of 8 characters, and an extension component of 3 characters. The DOS file manager will display this multiple component name with a period character between them, since period is an illegal filename character in a DOS filename and this follows the naming convention used in DOS when specifying filenames.

The DOS file system is a convenient interchange mechanism when a network is not available. It is not intended to be a high performance file system. Not all of the DG/UX system calls will operate on files from the DOS file system. They will return the `errno` EOPNOTSUPP if they do not operation on DOS files. These will be calls such as link, readlink, symlink, file locking calls, chmod, chown, chgrp, dg_unbuffered_write and dg_unbuffered_read.

DOS filesystems are not exportable over NFS.

If you remove the floppy from the drive before you umount the floppy, the system will prompt you for that floppy when you try to access the floppy drive again. It will require you to put in the correct floppy before allowing you access again. You can, however, umount the floppy without the floppy being present.

In order to use the DOS file manager, you must configure the dfm() pseudo device into your kernel.

```
sd(insc(),*)
st(insc(),*)
inen()
loop()
pmt()
prf()
meter()
dfm()       # this is the line that must be added.
```

Once the kernel is built and running, you may use the mount(1M) command to add the DOS floppy to the UNIX file system hierarchy.

```
mount -t dos /dev/pdsk/4 /pdd/floppy
```

The special device mentioned in the mount command is the block special representation of the floppy device in /dev/pdsk. The type "dos" must be used with mount to route the mount request to the correct file manager.

You may add a line to the /etc/fstab file to have the mount occur when the system is brought up to init level 3.

```
/dev/pdsk/4 /pdd/floppy dos rw x 0
```

The umount(1M) command may be used to unmount the DOS floppy from the file system hierarchy

```
umount /pdd/floppy
```

You can create DOS formatted floppies with the mkfs(1M) command:

```
mkfs 720kb /dev/rpdsk/4
```

SEE ALSO

mkfs(1M), mount(1M), umount(1M), config(1M), fstab(4).

NAME

dumpcycle - dump cycle file for backups

DESCRIPTION

The file `/etc/sysadm/dumpcycle` describes the cycle used by `admbackup`(1M) for creating backups of file systems.

Each line of the file describes the backup to be performed for a given day (excluding weekends). Each line contains four fields like this:

```
pattern  dump-level  multi-dump-flag  description
```

By default, the first line of the file is as follows:

```
[dwm]          0          n          Monthly Set
```

`admbackup`(1M) compares the *pattern* field against the string in the fifth field of the `fstab`(4) entry for each file system mounted on the local machine. Each file system whose entry matches the *pattern* for the current day will be backed up.

The *dump-level* field indicates the level of the backup to be performed. If *dump-level* is `0`, all files in the file system are backed up. If *dump-level* is greater than `0`, all files which have changed since a previous, lower-level dump was performed will be backed up (see `dump2`(1M)).

The *multi-dump-flag* field exists for compatibility with previous releases, and will be removed in a future release.

The *description* field is a text string which is presented to the operator as a suggested label for this day's backup. Typically, this field is used on the paper labels applied to backup tapes.

One and only one line of the dumpcycle file must contain the character "@" at the beginning of the line. This character indicates the current position in the cycle. This position may be modified by using the **admdumpcycle −o position** command.

EXAMPLES

If a machine has a large-capacity tape drive attached to it, the administrator may choose to alter the cycle of backups so that more information is backed up every day. This may allow files to be restored more quickly.

The entries needed to perform a full backup once a week and a backup of all changed files each day of the week would look similar to this:

```
@[dwm]      0        n        Monthly Set
[d]         1        n        Week 1 - Monday Set
[d]         1        n        Week 1 - Tuesday Set
[d]         1        n        Week 1 - Wednesday Set
[d]         1        n        Week 1 - Thursday Set
[dw]        0        n        Week 1 - Weekly Set
[d]         1        n        Week 2 - Monday Set
[d]         1        n        Week 2 - Tuesday Set
[d]         1        n        Week 2 - Wednesday Set
[d]         1        n        Week 2 - Thursday Set
```

and so on.

FILES

`/etc/sysadm/dumpcycle`

SEE ALSO

`admbackup`(1M), `admdumpcycle`(1M), `dump2`(1M), `fstab`(4).

NAME

`ftpd.deny` – File to disallow incoming FTP sessions for particular login names

DESCRIPTION

The `ftpd.deny` file, located in the `/etc` directory, gives system administrators the ability to control File Transfer Protocol (FTP) access to their systems.

If an administrator wants to deny `ftp` access to a particular user, that user's login name should be entered into the `ftpd.deny` file. When that user attempts to `ftp` into the system, the ftp daemon (`ftpd`) scans the `ftpd.deny` file, finds the user's login name, and denies access to that user.

The anonymous login name is checked before the `ftpd.deny` file, so it is not possible to disallow anonymous access using this file. If you want to disallow anonymous access, you should remove the ftp username from the `/etc/passwd` file.

Entries in the file should be user login names, each name on a separate line. Names should not be preceded or followed by white space. Comment lines must have a "#" at the beginning of the line.

FILES

`/etc/ftpd.deny`

SEE ALSO

`ftp`(1C), `ftpd`(1M), `passwd`(4).

NAME

gateways – database for routed

DESCRIPTION

When you start routed, it reads the /etc/gateways file to specify routing gateways. The file consists of a series of lines, each in the following format:

[net | host] *n1* gateway *n2* metric *val* [passive | active | external]

The net or host keyword indicates if the route is to a network or specific host.

n1 is the name of the destination network or host. This may be a symbolic name located in /etc/networks or /etc/hosts, or an Internet address specified in "dot" notation; see inet(3N).

n2 is the name or address of the gateway to which messages should be forwarded.

val is a metric indicating the hop count to the destination host or network.

One of the keywords passive, active or external indicates if the gateway should be treated as passive or active or whether the gateway is external to the scope of the routed protocol.

Gateways specified in /etc/gateways should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange routing information (that is, they should have a routed process running on the machine). Passive gateways are maintained in the routing tables forever and information regarding their existence is included in any routing information transmitted. Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway and if no routing information is received for a period of the time, the associated route is deleted. External gateways are also passive, but are not placed in the kernel routing table nor are they included in routing updates. The function of external entries is to inform routed that another routing process will install such a route, and that alternate routes to that destination should not be installed. Such entries are only required when both routers may learn of routes to the same destination.

FILES

/etc/gateways

SEE ALSO

routed(1M).

NAME

`hosts.equiv` – file format list of trusted hosts database

DESCRIPTION

The `hosts.equiv` file, located in the `/etc` directory, gives the system administrator the ability to control remote access. The `.rhosts` file, located in a local user's home directory, gives each user the ability to control remote access.

When a remote user on a remote host makes an `rlogin(1C)` or `remsh(1C)`(formerly `rsh`) request, the `ruserok(rcmd(3X))` function scans the trusted host entries in `hosts.equiv` and `.rhosts` to determine if the remote user on the remote host is trusted. If trusted, then the user is allowed remote access; `rlogin` does not prompt for a password, and `remsh` executes. Otherwise, `rlogin` prompts for a password, and `remsh` does not execute.

The `ruserok` function allows remote access if either `hosts.equiv` or `.rhosts` allows remote access. If `hosts.equiv` denies remote access but `.rhosts` allows remote access, then remote access is allowed. If the local user is root(user id is 0), then only `.rhosts` is searched.

A trusted host entry can allow remote access, deny remote access, or make no decision. The `ruserok` function scans each file linearly, allowing or denying remote access based on the first entry that allows or denies remote access. If no entry allows or denies remote access, then remote access is denied. If an entry in `hosts.equiv` denies remote access before another entry in `hosts.equiv` allows remote access, then remote access is denied by `hosts.equiv`. However, remote access can still be allowed by `.rhosts`.

The `hosts.equiv` and `.rhosts` files have one trusted host entry per line. A trusted host entry consists of a hostname expression and an optional username expression, delimited by any number of blanks and/or tab characters. An entry allows remote access if both the hostname expression and the username expression allow remote access. An entry denies remote access if the hostname expression denies remote access, or the hostname expression allows remote access but the username expression denies remote access.

The hostname expression can be any of the following:

`+` Allow remote access to all hosts.

`+@`*groupname*
: Allow remote access to all hosts in the `netgroup(4)` *groupname*.

`-@`*groupname*
: Deny remote access to all hosts in the `netgroup` *groupname*.

hostname
: Allow remote access to the host named *hostname*.

-hostname
: Deny remote access to the host named *hostname*.

If the domain name system is used, a separate entry must be made for the simple and the fully–qualified hostnames. For example, *sales* and *sales.hq.acme.com*. If the remote host has more than one interface, a separate entry must be made for each of the host's interfaces. For example, *accounting* and *accounting-alt*.

The username expression can be any of the following:

`+` Allow remote access to all users.

+@groupname
Allow remote access to all users in the `netgroup` *groupname*.

-@groupname
Deny remote access to all users in the `netgroup` *groupname*.

username
Allow remote access to the user named *username*.

-username
Deny remote access to the user named *username*.

If the username expression is omitted, then remote access is allowed if the remote username and the local username are the same. For example, the remote user `bob` must log in to the local host as `bob`, not as `sally` or `billy`.

EXAMPLE

The following is an example `hosts.equiv` file:

```
+                           +@engineering
+                           -@marketing
+                           -billy
sales
sales.org.acme.com
sales-alt
sales-alt.org.acme.com
-accounting
-accounting.org.acme.com
qa                          sally
qa.org.acme.com                     sally
```

The first entry allows remote access to all users in the `netgroup` *engineering*. The next entry denies remote access to all users in the `netgroup` *marketing*. The third entry denies remote access to *billy* from all hosts. If *billy* were in the `netgroup` *engineering*, he would have already been allowed remote access with the first entry. The *sales* entries allow remote access to users on the host *sales* who log in with the same remote and local usernames. The host *sales* has two interfaces named *sales* and *sales-alt*. Since the domain name system is being used, there are separate entries for the simple name, *sales*, and the fully-qualified name, *sales.org.acme.com*. The next two entries deny remote access to users on the host *accounting*. The last two entries allow remote access to *sally* from host *qa*.

FILES

`/etc/hosts.equiv`
`/.rhosts`

SEE ALSO

`hostname`(1C), `remsh`(1C), `rlogin`(1C), `rcmd`(3X), `netgroup`(4).

NAME

`inetd.conf` – Internet servers database

DESCRIPTION

The `inetd.conf` file contains the list of servers that `inetd`(1M) invokes when it receives a request from the network. Each server entry is composed of a single line of the form:

service-name type protocol wait-status uid server-program server-arguments

You can separate fields with spaces or TAB characters. A '#' (pound-sign) indicates the beginning of a comment; characters up to the end of the line are not interpreted by routines that search this file.

service-name is the name of a valid service listed in the file `/etc/services`. For RPC services, the value of the *service-name* field consists of the RPC service name, followed by a slash and either a version number or a range of version numbers.

type can be one of:

`stream` for a stream socket,

`dgram` for a datagram socket,

`raw` for a raw socket,

`tli` for a Transport Layer Interface (TLI) endpoint.

protocol must be a recognized protocol listed in the file `/etc/protocols`. For RPC services, the field consists of the string rpc followed by a slash and the name of the protocol (for example, `rpc/udp` for an RPC service using the UDP protocol as a transport mechanism).

For type `tli` endpoints, this field represents the source provider. If the source provider does not begin with `/`, it is assumed to be in `/dev`.

wait-status is `nowait` for all but single-threaded datagram servers — servers which do not release the socket until a timeout occurs. These must have the status `wait`.

uid is the user ID under which the server should run. This allows servers to run with access privileges other than those for root.

server-program is either the pathname of a server program to be invoked by `inetd` to perform the requested service, or the value `internal` if `inetd` itself provides the service.

server-arguments If a server must be invoked with command-line arguments, the entire command line (including argument 0) must appear in this field (which consists of all remaining words in the entry). If the server expects `inetd` to pass it the address of its peer (for compatibility with 4.2BSD executable daemons), then the first argument to the command should be specified as '`%A`'.

FILES

```
/etc/inetd.conf
/etc/services
/etc/protocols
```

SEE ALSO

inetd(1M), services(4).

NAME

mailcnfg - initialization information for mail and rmail

DESCRIPTION

The /etc/mail/mailcnfg file contains initialization information for the mail and rmail commands. Each entry in mailcnfg consists of a line of the form

Keyword = Value

Leading whitespace, whitespace surrounding the equal sign, and trailing whitespace is ignored. *Keyword* may not contain embedded whitespace, but whitespace may appear within *Value*. Undefined keywords or badly formed entries are silently ignored.

Keyword Definitions

DEBUG — Takes the same values as the -x invocation option of mail. This provides a way of setting a system-wide debug/tracing level. Typically DEBUG is set to a value of 2, which provides minimal diagnostics useful for debugging mail and rmail failures. The value of the -x mail invocation option will override any specification of DEBUG in mailcnfg.

CLUSTER — To identify a closely coupled set of systems by one name to all other systems, set *Value* to the cluster name. This string is used to supply the **...remote from...** information on the **From** header line rather than the system nodename returned by uname(2).

FAILSAFE — In the event that the /var/mail directory is accessed via RFS or NFS within a cluster (see CLUSTER above), provisions must be made to allow for the directory not being available when local mail is to be delivered (remote system crash, RFS or NFS problems, etc.). *Value* is a string that indicates where to forward the current message for delivery. Typically this is the remote system that actually *owns* /var/mail. In this way, the message is queued for delivery to that system when it becomes available. For example, assume a cluster of systems (sysa, sysb, sysc) where /var/mail is physically mounted on sysc and made available to the other machines via RFS or NFS. If sysc were to crash, the RFS/NFS-accessible /var/mail would become unavailable and local deliveries of mail would go to /var/mail on the local system. When /var/mail is re-mounted via RFS/NFS, all messages deposited in the local directory would be hidden and essentially lost. To prevent this, if FAILSAFE is defined in mailcnfg, mail and rmail check for the existence of /var/mail/:saved, a required subdirectory. If this subdirectory does not exist, mail assumes that the RFS/NFS-accessible /var/mail is not available and invokes the failsafe mechanism of automatically forwarding the message to *Value*. In this example *Value* would be sysc!%n. The *%n* keyword is expanded to be the recipient name [see mail(1) for details] and thus the message would be forwarded to sysc!*recipient_name*. Because sysc is not available, the message remains on the local system until sysc is available, and then sent there for delivery.

DEL_EMPTY_MFILE — If not specified, the default action of mail and rmail is to delete empty mailfiles if the permissions are 0660 and to retain

empty mailfiles if the permissions are anything else. If *Value* is `yes`, empty mailfiles are always deleted, regardless of file permissions. If *Value* is `no`, empty mailfiles are never deleted.

`DOMAIN` — This string is used to supply the system domain name in place of the domain name returned by `getdomainame`(3).

`SMARTERHOST` — This string may be set to a smarter host which may be referenced within the mail surrogate file via `%X`.

`%`*mailsurr_keyword* — As described in `mailsurr`(4), certain pre-defined single letter keywords are textually substituted in surrogate command fields before they are executed. While none of the predefined keywords may be changed in meaning, new ones may be defined to provide a shorthand notation for long strings (such as `/usr/lib/mail/surrcmd`) which may appear repeatedly within the `mailsurr` file. Upper case letters are reserved for future use and will be ignored if encountered here.

FILES

`/etc/mail/mailcnfg`
`/etc/mail/mailsurr`
`/var/mail/:saved`
`/usr/lib/mail/surrcmd`

SEE ALSO

`mailsurr`(4)
`mail`(1) in the *User's Reference Manual*
`uname`(2), `getdomainame`(3) in the *Programmer's Reference Manual*

NOTES

If `/var/mail` is accessed via RFS or NFS and the subdirectory `/var/mail/:saved` is not removed from the local system, the `FAILSAFE` mechanism will be subverted.

NAME

`mailsurr` – surrogate commands for routing and transport of mail

DESCRIPTION

The `mailsurr` file contains routing and transport surrogate commands used by the `mail` command. Each entry in `mailsurr` has three whitespace-separated, single quote delimited fields:

'sender' 'recipient' 'command'

or a line that begins

`Defaults:`

Entries and fields may span multiple lines, but leading whitespace on field continuation lines is ignored. Fields must be less than 1024 characters long after expansion (see below).

The sender and recipient fields are regular expressions. If the sender and recipient fields match those of the message currently being processed, the associated command is invoked.

The *command* field may have one of the following five forms:

`A[ccept]`
`D[eny]`
`T[ranslate] R=[|]`*string*
`<` `S=...;C=...;F=...;` *command*
`>` *command*

Regular Expressions

The sender and recipient fields are composed of regular expressions (REs) which are digested by the `regexp`(5) `compile` and `advance` procedures in the C library. The regular expressions matched are those from `ed`(1), with simple parentheses `()` playing the role of `\(\)` and the addition of the `+` and `?` operators from `egrep`(1). Any single quotes embedded within the REs *must* be escaped by prepending them with a backslash or the RE is not interpreted properly.

The `mail` command prepends a circumflex (`^`) to the start and appends a dollar sign (`$`) to the end of each RE so that it matches the entire string. Therefore it would be an error to use *^RE$* in the sender and recipient fields. To provide case insensitivity, all REs are converted to lower case before compilation, and all sender and recipient information is converted to lower case before comparison. This conversion is done only for the purposes of RE pattern matching; the information contained within the message's header is *not* modified.

The sub-expression pattern matching capabilities of `regexp` may be used in the command field, that is, `(...)`, where $1 \leq n \leq 9$. Any occurrences of `\\`*n* in the replacement string are themselves replaced by the corresponding `(...)` substring in the matched pattern. The sub-expression fields from both the sender and recipient fields are accessible, with the fields numbered 1 to 9 from left to right.

Accept and Deny Commands

`Accept` instructs `rmail` to continue its processing with the `mailsurr` file, but to ignore any subsequent matching `Deny`. That is, unconditionally accept this message for delivery processing. `Deny` instructs `rmail` to stop processing the `mailsurr` file and to send a negative delivery notification to the originator of the message. Whichever is encountered first takes precedence.

Translate Command

`Translate` allows optional on-the-fly translation of recipient address information. The *recipient* replacement string is specified as `R=`*string*.

For example, given a command line of the form

```
'.+' '([^!]+)@(.+)\.EUO\.ATT\.com' 'Translate R=attmail!\\2!\\1'
```

and a recipient address of `rob@sysa.EUO.ATT.COM` the resulting recipient address would be `attmail!sysa!rob`.

Should the first character after the equal sign be a '|', the remainder of the string is taken as a command line to be directly executed by `rmail`. If any `sh`(1) syntax is required (metacharacters, redirection, etc.), then the surrogate command must be of the form:

`sh -c "`*shell command line...*`"`

Special care must be taken to escape properly any embedded back-slashes and single or double quotes, since `rmail` uses double quoting to group whitespace delimited fields that are meant to be considered as a single argument to `execl`(2). It is assumed that the executed command will write one or more replacement strings on `stdout`, one per line. If more than one line is returned, each is assumed to be a different recipient for the message. This mechanism is useful for mailing list expansions. As stated above, any occurrences of `\\`*n* are replaced by the appropriate substring *before* the command is executed. If the invoked command does not return at least one replacement string (no output or just a newline), the original string is *not* modified. For example, the command line

```
'.+'  '(.+)' 'Translate R=|/usr/bin/findpath \\1'
```

allows local routing decisions to be made.

If the recipient address string is modified, `mailsurr` is rescanned from the beginning with the new address(es), and any prior determination of `Accept` (see above) is discarded.

`<` *command*

The intent of a `<` command is that it is invoked as part of the transport and delivery mechanism, with the ready-for-delivery message available to the command at its standard input. As such, there are three conditions possible when the command exits:

Success — The command successfully delivered the message. What actually constitutes successful delivery may be different within the context of different surrogates. The `rmail` process assumes that no more processing is required for the message for the current recipient.

Continue — The command performed some function (logging remote message traffic, for example) but did not do what would be considered message delivery. The `rmail` process continues to scan the `mailsurr` file looking for some other delivery mechanism.

Failure — The command encountered some catastrophic failure. The `rmail` process stops processing the message and sends to the originator of the message a non-delivery notification that includes any `stdout` and `stderr` output generated by the command.

The semantics of the `<` command field in the `mailsurr` file allow the specification of exit codes that constitute success, continue, and failure for each surrogate command individually. The syntax of the exit state specification is:

< WS [*exit_state_id*=*ec*[,*ec*[,...]];][*exit_state_id*=*ec*[,*ec*[,...]];
[...]]] WS *surrogate_cmd_line*

WS is whitespace. *exit_state_id* can have the value `S`, `C`, or `F`. *exit_state_id*s can be specified in any order. *ec* can be:

any integer $0 \leq n \leq 255$ [Negative exit values are not possible. See `exit`(2) and `wait`(2).]

a range of integers of the form *lower_limit*–*upper_limit* where the limits are $\geq$ 0 and $\leq$ 255, and

`*`, which implies *anything*

For example, a command field of the form:

'< S=1-5,99;C=0,12;F=*; *command* %R'

indicates that exit values of 1 through 5, and 99, are to be considered success, values of 0 (zero) and 12 indicate continue, and that anything else implies failure. If not explicitly supplied, default settings are `S=0;C=*;`.

It may be possible for ambiguous entries to exist if two exit states have the same value, for example, `S=12,23;C=*;F=23,52`; or `S=*;C=9;F=*;`. To account for this, `rmail` looks for *explicit* exit values (that is, *not* "*") in order of success, continue, failure. Not finding an explicit match, `rmail` then scans for "*" in the same order.

It is possible to eliminate an exit state completely by setting that state's value to an impossible number. Since exit values must be between 0 and 255 (inclusive), a value of 256 is a good one to use. For example, if you had a surrogate command that was to log all message traffic, a `mailsurr` entry of

```
'(.+)' '(.+)' '<S=256;C=*; /usr/lib/mail/surrcmd/logger \\1 \\2'
```

would always indicate continue.

Surrogate commands are executed by `rmail` directly. If any shell syntax is required (metacharacters, redirection, etc.), then the surrogate command must be of the form:

`sh -c` "*shell command line...*"

Special care must be taken to properly escape any embedded back-slashes and other characters special to the shell as stated in the "Translate" section above.

If there are no matching < commands, or all matching < commands exit with a continue indication, `rmail` attempts to deliver the message itself by assuming that the recipient is local and delivering the message to `/var/mail/`*recipient*.

> command

The intent of a > command is that it is invoked *after* a successful delivery to do any post-delivery processing that may be required. Matching > commands are executed only if some < command indicates a successful delivery (see the previous section) or local delivery processing is successful. The `mailsurr` file is rescanned and all matching > commands, not just those following the successful < command, are executed in order. The exit status of an > command is ignored.

Defaults: Line

The default settings may be redefined by creating a separate line in the `mailsurr` file of the form

`Defaults: [S=...;][C=...;][F=...;]`

`Defaults:` lines are honored and the indicated default values redefined when the line is encountered during the normal processing of the `mailsurr` file. Therefore, to redefine the defaults globally, the `Defaults:` line should be the first line in the file. It is possible to have multiple `Defaults:` lines in the `mailsurr` file, where each subsequent line overrides the previous one.

Surrogate Command Keyword Replacement.

Certain special sequences are textually-substituted in surrogate commands before they are invoked:

`%n`	the recipient's full name.
`%R`	the full return path to the originator (useful for sending replies, delivery failure notifications, etc.)
`%c`	value of the **`Content-Type:`** header line if present.
`%C`	"**`text`**" or "**`binary`**", depending on an actual scan of the content. This is independent of the value of any **`Content-Type`** header line encountered (useful when calling `ckbinarsys`.)
`%S`	the value of the **`Subject:`** header line, if present.
`%l`	value of the **`Content-Length:`** header line.
`%L`	the local system name. This will be either `CLUSTER` from `mailcnfg` or the value returned by `uname`.
`%U`	the local system name, as returned by `uname`.
`%X`	the value of `SMARTERHOST` in `mailcnfg`.
`%D`	the local domain name. This will be either `DOMAIN` from `mailcnfg`, or the value returned by `getdomainame`.
`\\`*n*	as described above, the corresponding (...) substring in the matched patterns. This implies that the `regexp` limitation of 9 substrings is applied to the sender and recipient REs collectively.
`%`*keywords*	Other keywords as specified in `/etc/mail/mailcnfg`. See `mailcnfg`(4).

The sequences `%L`, `%U`, `%D`, and `%`*keywords* are permitted within the sender and recipient fields as well as in the command fields.

An example of the `mailsurr` entry that replaces the `uux` "built-in" of previous versions of `rmail` is:

```
'.+'   '([^@!]+)!(.+)'    '< /usr/bin/uux - \\1!rmail (\\2)'
```

Mail Surrogate Examples

Some examples of mail surrogates include the distribution of message-waiting notifications to LAN-based recipients and lighting Message-Waiting Lamps, the ability to mail output to printers, and the logging of all `rmail` requests between remote systems (messages passing through the local system). The following is a sample `mailsurr` file:

```
#
# Some common remote mail surrogates follow. To activate any
# or all of them, remove the `#' (comment indicators) from
# the beginning of the appropriate lines. Remember that they
# will be tried in the order they are encountered in the file,
# so put preferred surrogates first.

#       Prevent all shell meta-characters
'.+'    '.*[`;&|^<>()].*'       'Deny'

#       Map all names of the form local-machine!user -> user
```

```
'.+'    '%L!(.+)'                'Translate R=\1'

#       Map all names of the form uname!user -> user
#       Must be turned on when using mail in a cluster environment.
#'.+'   '%U!(.+)'                'Translate R=\1'

#       Map all names of the form user@host -> host!user
'.+'    '([^!@]+)@(.+)'          'Translate R=\2!\1'

#       Map all names of the form host.uucp!user -> host!user
'.+'    '([^!@]+)\.uucp!(.+)'    'Translate R=\1!\2'

#       Map all names of the form host.local-domain!user -> host!user
#       DOMAIN= within /etc/mail/mailcnfg will override getdomainname(3).
'.+'    '([^!@]+)%D!(.+)'        'Translate R=\1!\2'

#       Allow access to 'attmail' from remote system 'sysa'
'sysa!.*' 'attmail!.+'   'Accept'

#       Deny access to 'attmail' from all other remotes
'.+!.+' 'attmail!.+'   'Deny'

#       Send mail for 'laser' to attached laser printer
#       Make certain that failures are reported via return mail.
'.+'    'laser'   '< S=0;F=*; lp -dlaser'

#       Run all local names through the mail alias processor
#
'.+'    '[^!@]+'         'Translate R=|/usr/bin/mailalias %n'

#       For remote mail via nusend
#'.+'   '([^!]+)!(.+)'   '< /usr/bin/nusend -d \\1 -s -e -!"rmail \\2" -'

#       For remote mail via usend
'.+'    '([^!]+)!(.+)'
      '< /usr/bin/usend -s -d\\1 -uNoLogin -!"rmail \\2" - '

#       For remote mail via uucp
'.+'    '([^!@]+)!.+'   '<S=256;C=0;
                /usr/lib/mail/surrcmd/ckbinarsys -t %C -s \\1'
'.+'    '([^!@]+)!(.+)'   '< /usr/bin/uux - \\1!rmail (\\2)'

#       For remote mail via smtp
#'.+'   '([^!@]+)!(.+)'         '< /usr/lib/mail/surrcmd/smtpqer %R %n'

#       If none of the above work, then let a router change the address.
#'.+'   '.*[!@].*'      'Translate R=| /usr/lib/mail/surrcmd/smail -A %n'

#       If none of the above work, then ship remote mail off to a smarter host.
#       Make certain that SMARTERHOST= is defined within /etc/mail/mailcnfg.
#'.+'   '.*[!@].*'              'Translate R=%X!%n'

#       Log successful message deliveries
```

'(.+)' '(.+)' '>/usr/lib/mail/surrcmd/logger \1 \2'

Note that invoking `mail` to read mail does not involve the `mailsurr` file or any surrogate processing.

Security

Surrogate commands execute with the permissions of `rmail` (user ID of the invoker, group ID of mail). This allows surrogate commands to validate themselves, checking that their effective group ID was `mail` at invocation time. This requires that all additions to `mailsurr` be scrutinized before insertion to prevent any unauthorized access to users' mail files. All surrogate commands are executed with the path `/usr/lib/mail/surrcmd:/usr/bin`.

Debugging New `mailsurr` Entries

To debug `mailsurr` files, use the `-T` option of the `mail` command. The `-T` option requires an argument that is taken as the pathname of a test `mailsurr` file. If null (as in `-T ""`), the system `mailsurr` file is used. Enter

`mail -T` *test_file recipient*

and some trivial message (like "`testing`"), followed by a line with either just a dot ("`.`") or a cntl-D. The result of using the `-T` option is displayed on standard output and shows the inputs and resulting transformations as `mailsurr` is processed by the `mail` command for the indicated *recipient*.

Mail messages will never be sent or delivered when using the `-T` option.

FILES

`/etc/mail/mailsurr`	
`/usr/lib/mail/surrcmd/*`	surrogate commands
`/etc/mail/mailcnfg`	initialization information for `mail`

SEE ALSO

`ckbinarsys`(1M), `mailcnfg`(4)
`mail`(1), `sh`(1), `uux`(1), `ed`(1), `egrep`(1), in the *User's Reference Manual*
`exec`(2), `exit`(2), `wait`(2), `popen`(3), `regexp`(5), `getdomainname`(3) in the *Programmer's Reference Manual*

NOTES

It would be unwise to install new entries into the system `mailsurr` file without verifying at least their syntactical correctness via '`mail -T` ...' as described above.

NAME

pmterrtab – table of equivalent error numbers

DESCRIPTION

The file /etc/pmterrtab contains an ASCII table of error numbers that are semantically equivalent among eight (8) different operating systems. The system administrator can modify its contents with a text editor. It is read by programs such as pmtd(1) that need to translate error numbers from other operating systems into DG/UX errno values.

The pmterrtab file must be in the following format:

First Line: *os1 os2 os3 os4 os5 os6 os7 os8*

where *os** is a name for the particular operating system. For example, dg or sun may be used.

Other Lines: *err1 err2 err3 err4 err5 err6 err7 err8*

where *err** is an error number of the operating system for that column. For Unix systems, errno values are used as error numbers.

Fields within the same line must be separated with spaces or tabs. Lines beginning with a # are ignored. They may used to add comments.

It is the responsibility of the system administrator to maintain this file and keep it up to date relative to the various operating systems' new releases and error numbers.

Some operating system vendors use the same error numbers. For example, dg and any BCS compliant system would use the same error numbers.

A Unix operating system vendor will probably use the same error numbers as att, sun, or dg. Error number sharing makes it easy to maintain the table because you do not have to fill in the values for new vendors so long as the vendor uses the same error numbers as a vendor already in the table.

If there is no equivalent error number for a particular operating system, a value of 0 should be placed in that entry.

FILES

/etc/pmterrtab Table of equivalent error numbers.
/usr/include/sys/errno.hTable describing DG/UX error numbers.

EXAMPLES

```
$ cat /etc/pmterrtab.example
# operating systems:
#--- ----  ----  ----  ----  ----  ----  ----
dg   sun   att   xxx   xxx   xxx   xxx   xxx
# error numbers :
#--- ----  ----  ----  ----  ----  ----  ----
2    2     2     0     0     0     0     0
38   0     38    0     0     0     0     0
45   78    45    0     0     0     0     0
130  38    0     0     0     0     0     0
```

Although this example file is not complete, it demonstrates two importants points: First, error numbers may vary widely from one operating system to another, or they may be the same. Second, not every operating system has error numbers that correspond to another's.

SEE ALSO

pmtd(1M), pmttapetab(4M).

CAVEATS

The pmtd(1) server expects a column named dg to exist in this file.

NAME

pmttapetab – table of remote tape devices

DESCRIPTION

The file `/etc/pmttapetab` contains an ASCII table describing magnetic tape devices on remote hosts. It associates local pseudo tape devices with these remote tape devices. The system administrator can modify its contents with a text editor. It is read by the `pmtd`(1M) daemon, which performs remote tape access across a network.

The file consists of a number of lines of the following format:

pseudo host os real rmt_dir block cache

where:

pseudo the local pseudo tape device. It should be the name of the particular special file created at boottime in the `/dev/pmt` directory. This is a filename.

host the name of the remote host machine. This machine MUST have the name of the client machine (the one running `pmtd`(1M)) in its `/etc/hosts.equiv` file.

os the name of the operating system running on *host*. This should be one of the operating system names specified in `/etc/pmterrtab`.

real the remote real tape device, that is, the device name on *host*. Only character special devices should be named. This is a full pathname.

rmt_dir the name of the directory on *host* that contains the `rmt`(1) daemon executable. This is a full pathname.

block a modulo block size that the remote hardware requires. The size of data transfers must be an integer multiple of this value. A value of 0 indicates that the remote hardware has no block size requirement.

cache a flag specifying that data transfers should be cached. Specify `Y` to cache and `N` not to cache.

You should use spaces or tabs to separate fields. Lines beginning with a `#` are ignored. They may be used to add comments. If you need to use spaces or tabs in the *real* field, surround the contents of the field with double quotes.

It is the responsibility of the system administrator to maintain this file and keep it up to date.

EXAMPLE

For example, assume `/etc/pmttapetab` contains the following entry:

```
0n      atlanta      dg      /dev/rmt/0n      /etc      512      N
```

In this case, entering the following command line would access the no rewind rmt 0 device on the host atlanta:

```
tar -xvf /dev/pmt/0n
```

`sysadm` considers pmt devices to be valid input mediums.

FILES

`/etc/pmttapetab`	Table with information about remote tape devices.
`/etc/pmterrtab`	Table of equivalent error numbers.
`/etc/hosts.equiv`	List of trusted host machines.

SEE ALSO

pmtd(1M), rmt(1M), hosts.equiv(4M), pmterrtab(4M), pmt(7), rmt(7).

CAVEATS

The real device name on the remote host must be a character special device. Typically these are kept in the host's /dev/rmt directory. Programs that read /etc/pmttapetab will assume the real tape devices listed are character special.

No-rewind-on-close *pseudo* entries should only be paired with no-rewind-on-close *real* entries in the /etc/pmttapetab file. The same follows for rewind-on-close entries.

NAME

`resolv.conf` – configuration file for name server routines

DESCRIPTION

The resolver configuration file contains information that is read by the resolver routines the first time they are invoked in a process. The file is designed to be human readable and contains a list of name-value pairs that provide various types of resolver information.

The different configuration options are:

`nameserver` *address* The Internet address (in dot notation) of a name server that the resolver should query. At least one name server should be listed. Up to MAXNS (currently 3) name servers may be listed, in that case the resolver library queries tries them in the order listed. (The algorithm used is to try a name server, and if the query times out, try the next, until out of name servers, then repeat trying all the name servers until a maximum number of retries are made).

`domain` *name* The default domain to append to names that do not have a dot in them. This defaults to the domain set by the `hostname(1C)` command.

`address` *address* An Internet address (in dot notation) of any preferred networks. The list of addresses returned by the resolver will be sorted to put any addresses on this network before any others.

The name value pair must appear on a single line, and the keyword (for instance, `nameserver`) must start the line. The value follows the keyword, separated by white space.

FILES

`/etc/resolv.conf`

SEE ALSO

`hostname`(1C), `named`(1M), `gethostent`(3N), `resolver`(3C).

NAME

snmpd.config, snmpd.communities, snmpd.trap_communities - SNMP configuration files

DESCRIPTION

Use these files to configure snmpd for your system. When snmpd is started, it reads the configuration files and begins servicing requests from the network. If snmpd recieves a SIGHUP signal, it re-reads the configuration files and then continues processing requests.

snmpd.config

Use this file to override the default values for some objects. You should use either the sysadm or admsnmpobject commands to modify this file. The format for this file is:

object=value

where object may be one of: sysDescr, sysObjectID, sysContact, sysLocation, or sysName. In most cases only the sysContact and sysLocation need to be specified in this file, because snmpd provides reasonable defaults for the other objects.

snmpd.communities

Use this file to define the community strings, host addresses, and access levels recognized by the agent. You should use either the sysadm or admsnmpcommunity commands to modify this file. There is a limit of 64 entries in snmpd.communities file and they have the following format:

community host access

where:

community	is an ASCII string of up to 64 characters specifing a community name to be recognized by the agent.
host	is either a hostname or Internet address from which the agent is willing to accept queries for this *community*.
access	is one of: READ, for read-only access, WRITE, for read-write access, or NONE for no access.

snmpd.trap_communities

Use this file to define the list of communities, host addresses, and port numbers to which the agent sends traps. You should use either the sysadm or admsnmptraps commands to modify this file. There is a limit of 64 entries in snmpd.trap_communities file and they have the following format:

community host port

where:

community	is an ASCII string of up to 64 characters specifying a community name to send with the trap message.

host is a hostname or Internet address to which the agent should send the trap.

port is the UDP port number, which is usually set to 162.

SEE ALSO

`admsnmpcommunity`(1M), `admsnmpobject`(1M), `admsnmptrap`(1M), `snmpd`(1M), `sysadm`(1M).

NAME

tcpip.params - TCP/IP network parameter database for rc scripts

DESCRIPTION

The tcpip.params(4M) file contains parameters for various commands invoked by the rc scripts to initialize the network.

The hostname(1C) command is run by the rc.tcpipport script. The following parameter is used by the hostname command:

hostname_ARG — The name you assign to the local host. For example, a host named *hostb* could be represented as follows:

```
hostname_ARG="hostb"
```

The hostid(1C) command is run by the rc.tcpipport script. The following parameter is used by the hostid command:

hostid_ARG — The ID is a hexadecimal number formed by the concatenation of the hexadecimal representation of the fields of the local host's internet address. For example, an internet address 85.0.0.31 would be represented by 1) converting each field to hexadecimal (which is 0x55.0x00.0x00.0x1f) and 2) concatenating these four fields (which is 0x5500001f). Therefore,

```
hostid_ARG="0x5500001f"
```

The ifconfig(1M) command is run by the rc.tcpipport script to start and stop network interfaces. Each line between the START_INTERFACE and STOP_INTERFACE delimeters contains a network interface entry. Each line contains a subset of the following parameters.

Parameter values for HOSTNAME, NETMASK, and BROADCAST can be expressed in Internet address dot notation or hexadecimal format. In addition, HOSTNAME can be expressed as a symbolic name. Any symbolic name used must be defined in the local /etc/hosts file for the name to be resolved correctly. The use of symbolic name references is recommended.

HOSTNAME — The name that associates an Internet address with the network interface to be configured. If a symbolic name is entered, the name must have an entry in the local /etc/hosts file.

All network interfaces require a value for HOSTNAME.

DEVICE — The name of the device to be configured.

For Ethernet(for example inen0, hken0), token ring(for example vitr0), and loopback(loop0) network interfaces, the device MUST have a corresponding entry in the /dev directory. When you add, modify, or delete the device name for Ethernet, token ring, or loopback network interfaces, you must reconfigure the kernel.

For IXE interfaces, there is only one entry in the /dev directory, namely /dev/ixe. This is because /dev/ixe is a cloneable device. When you bring up TCP/IP, the system creates a symbolic link in the /dev directory (for example,

/dev/ixe1) to the corresponding IXE template file to preserve the mapping between instances of the ixe device and template files.

All network interfaces require a value for DEVICE.

NETMASK — The network(subnet) mask assigned to the configured network interface. If no value is entered, the ifconfig command uses the default netmask which prevents subnetting.

Network interfaces on subnetted networks require a value for NETMASK.

BROADCAST — The broadcast Internet address assigned to the configured network interface. The network portion of the broadcast address must be the same as the network portion of the interface's Internet address. The host portion should be all ones (BSD 4.3 compatible) or all zeros (BSD 4.2 compatible).

Ethernet and token ring network interfaces require a value for BROADCAST.

LINK_PROTO — The data link level protocol to be employed by the configured network interface.

Ethernet network interfaces require a value for LINK_PROTO; specify the value ether unless you intend to use 802.3.

TEMPLATE — The IXE template file associated with the network interface.

IXE network interfaces require a value for TEMPLATE. The template filename is two to eleven characters in length. The template filename is not a full pathname; template files are located in the /usr/opt/x25/etc/template directory.

For example, the lines below define the following network interface entries:
the name *localhost* using device *loop0*,
the name *hostb* using device *hken0* on the subnetted network 128.222.8,
the name *hostb-alt* using device *hken1* on the subnetted network 128.222.3,
the name *hostb-ixe* using device *ixe0* with the template file *ixefile*, and
the name *hostb-ring* using device *vitr0* on the subnetted network 128.222.5.

```
      START_INTERFACE
      HOSTNAME=localhost DEVICE=loop0
      HOSTNAME=hostb DEVICE=hken0 NETMASK=0xffffff00 \
BROADCAST=128.222.8.255 LINK_PROTO=ether
      HOSTNAME=hostb-alt DEVICE=hken1 NETMASK=0xffffff00 \
BROADCAST=128.222.3.255 LINK_PROTO=ether
      HOSTNAME=hostb-ixe DEVICE=ixe0 TEMPLATE=ixefile
      HOSTNAME=hostb-ring DEVICE=vitr0 NETMASK=0xffffff00 \
BROADCAST=128.222.5.255
      STOP_INTERFACE
```

The route(1M) command is run by the rc.tcpipport script to add and delete routing table entries. Each line between the START_ROUTE and STOP_ROUTE delimeters contains a routing table entry. Each line contains a subset of the following parameters.

Parameter values for DESTINATION and GATEWAY can be expressed in symbolic name, Internet address dot notation, or hexadecimal format. Any symbolic name used must be defined in the local /etc/hosts or /etc/networks file for the name to be resolved correctly. The use of symbolic name references is recommended.

TYPE — Indicates whether the route is to a host or a network. The possible values are host and net.

If no value is specified, the route command will default the type of route.
If the DESTINATION's Internet address host part is all zeros, TYPE will default to net.
If the DESTINATION's Internet address host part is NOT all zeros, TYPE will default to host.

Routes to subnetted networks require a value for TYPE.

The use of the TYPE parameter is recommended for all routes.

DESTINATION — The hostname or network name of the destination of the route.
A default route may be entered by specifying 0 or default.

All routes require a value for DESTINATION.

GATEWAY — The hostname of the interface or gateway through which traffic is routed to DESTINATION.

All routes require a value for GATEWAY.

METRIC — Specifies either an interface route or a gateway route.
A value of 0 specifies an interface route.
A value of 1 specifies a gateway route.
The default value for METRIC is 0.

All gateway routes require a value for METRIC.

The use of the METRIC parameter is recommended for all routes.

For example, the lines below define the following routing table entries:
route traffic to host *far-host* through *router*,
route traffic to network *far-net* through *router*, and
route all other traffic through *gateway*.

```
START_ROUTE
TYPE=host DESTINATION=far-host GATEWAY=router METRIC=1
TYPE=net DESTINATION=far-net GATEWAY=router METRIC=1
TYPE=net DESTINATION=default GATEWAY=gateway METRIC=1
STOP_ROUTE
```

The hosts *far-host*, *router*, and *gateway* must have entries in the local `/etc/hosts` file. The network *far-net* must have an entry in the local `/etc/networks` file.

NOTE: When `ifconfig` configures a network interface, an interface route to the directly connected network is added. Therefore, interface routes are not required here.

The network daemons are started and stopped by the `rc.tcpipserv` script. Each line between the START_DAEMON and STOP_DAEMON delimeters contains a network daemon entry. Each line contains the `DAEMON_NAME` parameter. If the daemon requires arguments, the line contains the `DAEMON_ARGS` parameter.

`DAEMON_NAME`	Name of the executable daemon program located in `/usr/bin`.
`DAEMON_ARGS`	Agruments to pass the daemon when starting. The arguments MUST be enclosed by double quotes.

For example, the lines below define the following daemon entries:
the *inetd* daemon,
the *snmpd* daemon, and
the *smtp* daemon with argument *-q30m*.

```
START_DAEMON
DAEMON_NAME=inetd
DAEMON_NAME=snmpd
DAEMON_NAME=smtp DAEMON_ARGS="-q30m"
STOP_DAEMON
```

SEE ALSO

`hostname`(1C), `hostid`(1C), `ifconfig`(1M), `route`(1M), *Managing TCP/IP on the DG/UX System*.

NAME

ttydefs – terminal line settings information for `ttymon`

SYNOPSIS

`/etc/ttydefs`

DESCRIPTION

`/etc/ttydefs` is an administrative file that contains information used by `ttymon`(1M) to set up the speed and terminal settings (i.e., the line discipline) for a TTY port. The `sttydefs`(1M) command maintains the contents of this file.

Each set of speed and terminal settings is represented by a single line in the file, separated by colons into fields as described below. In addition, comment lines beginning with a pound sign (#) are supported; blank lines for readability are supported, too. The file begins with the following version number line:

```
# VERSION=1
```

The settings lines of the `ttydefs` file have the following format:

ttylabel : *initial-flags* : *final-flags* : *autobaud* : *nextlabel*

where the fields are as follows:

ttylabel — The string a TTY port monitor will attempt to match against the TTY port's *ttylabel* field in the port monitor's administrative file. It typically describes the speed at which the terminal is supposed to run, for example, `9600`.

initial-flags — The initial `termio`(7) settings to which the line discipline is to be set. For example, the system administrator can specify what the default ERASE and KILL characters will be. *initial-flags* must be specified in the syntax recognized by the `stty`(1) command.

final-flags — The final `termio`(7) settings to which the line discipline is to be set after a connection request has been made and immediately prior to invoking a port's service. *final-flags* must be specified in the same format as *initial-flags*.

autobaud — If the autobaud field contains the character 'A', autobaud will be enabled. Otherwise, autobaud will be disabled. If autobaud is enabled, the TTY port monitor will determine the line speed for the TTY port by analyzing carriage returns entered at the terminal. If autobaud is disabled, the hunt sequence (see *nextlabel* below) will be used for baud rate determination.

nextlabel — If the user indicates that the current terminal setting is not appropriate by sending a BREAK, the TTY port monitor searches for a `ttydefs` entry whose *ttylabel* field matches the *nextlabel* field. If a match is found, the port monitor uses that field as its *ttylabel* field. A series of speeds is often linked together in this way into a closed set called a "hunt sequence". For example, modem entry `M4800` may be linked to `M2400`, which in turn is linked to `M1200`, which is finally linked back to `M4800`.

FILES

`/etc/ttydefs` TTY definitions file

SEE ALSO

`stty`(1), `sttydefs`(1M), `ttymon`(1M), `termio`(7)

Managing the DG/UX System

NOTES

The contents of this file are subject to change in future releases. The version number (currently 1) will be updated in conjunction with any such changes.

NAME

ttysrch - directory search list for ttyname

DESCRIPTION

ttysrch is an optional file that is used by the ttyname library routine. This file contains the names of directories in /dev that contain terminal and terminal-related device files. The purpose of this file is to improve the performance of ttyname by indicating which subdirectories in /dev contain terminal-related device files and should be searched first. These subdirectory names must appear on separate lines and must begin with /dev. Those path names that do not begin with /dev will be ignored and a warning will be sent to the console. Blank lines (lines containing only white space) and lines beginning with the comment character "#" will be ignored. For each file listed (except for the special entry /dev), ttyname will recursively search through subdirectories looking for a match. If /dev appears in the ttysrch file, the /dev directory itself will be searched but there will not be a recursive search through its subdirectories.

When ttyname searches through the device files, it tries to find a file whose major/minor device number, file system identifier, and inode number match that of the file descriptor it was given as an argument. If a match is not found, it will settle for a match of just major/minor device and file system identifier, if one can be found. However, if the file descriptor is associated with a cloned device (see clone(7)), this algorithm does not work efficiently because the inode number of the device file associated with a clonable device will never match the inode number of the file descriptor that was returned by the open of that clonable device. To help with these situations, entries can be put into the /etc/ttysrch file to improve performance when cloned devices are used as terminals on a system (e.g. for remote login). However, this is only useful if the minor devices related to a cloned device are put into a subdirectory. (It is important to note that device files need not exist for cloned devices and if that is the case, ttyname will eventually fail.) For example if /dev/starlan is a cloned device, there could be a subdirectory /dev/slan that contains files 0, 1, 2, etc. that correspond to the minor devices of the starlan driver. An optional second field is used in the /etc/ttysrch file to indicate the matching criteria. This field is separated by white space (any combination of blanks or tabs). The letter M means major/minor device number, F means file system identifier, and I means inode number. If this field is not specified for an entry, the default is MFI which means try to match on all three. For cloned devices the field should be MF, which indicates that it is not necessary to match on the inode number.

Without the /etc/ttysrch file, ttyname will search the /dev directory by first looking in the directories /dev/term, /dev/pts, and /dev/xt. If a system has terminal devices installed in directories other than these, it may help performance if the ttysrch file is created and contains that list of directories.

EXAMPLE

A sample /etc/ttysrch file follows:

```
/dev/term    MFI
/dev/pts          MFI
/dev/xt           MFI
/dev/slan    MF
```

This file tells ttyname that it should first search through those directories listed and that when searching through the /dev/slan directory, if a file is encountered whose major/minor devices and file system identifier match that of the file descriptor argument to ttyname, this device name should be considered a match.

FILES

/etc/ttysrch

SEE ALSO

ttyname(3C), clone(7)

NAME

vtc.addrs - SYAC VTC configuration file

SYNOPSIS

/etc/tcload/vtc.addrs

DESCRIPTION

The vtc.addrs file contains network configuration information for SYAC VTC boards. The information contained in this file is communicated to the board when rc.tcload is run or when the SYAC board is reset.

The vtc.addrs file contains entries for VTC specific information and entries for tty specific information. Each tty entry in this file must be for a tty that is associated with a VTC, and tty entries must immediately follow the VTC entry with which the tty is associated. Fields in all entries are whitespace separated. There must be one VTC specific entry for each configured SYAC that is a VTC. Tty specific entries are optional.

The tty specific entries allow the system administrator to alter the behavior of ttys associated with a VTC. By default, ttys associated with a VTC answer telnet connections to the default Internet address for that VTC. Ttys can also be configured to answer connections for different Internet addresses. This behavior is useful when it is necessary to associate a specific terminal entry with a specific device via a telnet connection (see syac_ttyaddrs(1M)).

Tty specific entries also allow the system administrator to determine how telnet binary mode affects onboard input processing (see termio(7)). By default, onboard input processing is unaffected by the state of telnet binary mode, and can only be enabled or disabled by changing the line discipline settings (see termio(7)). Ttys associated with a VTC can be configured such that when the telnet connection is in telnet binary mode, onboard input processing is also disabled. Note, however, that input processing performed by the host for a tty associated with a VTC will always be unaffected by the state of telnet binary mode. Very rarely will an application require that onboard input processing be disabled when telnet binary mode is in effect, as input processing is normally controlled exclusively via the line discipline settings.

VTC-specific Entries

The VTC-specific entries have the following format:

<SYAC node> <Inet Addr> <BAddr> <Netmask> <Route Info>

The SYAC node field specifies the full pathname of a SYAC control node [see syac(7)], which must refer to a VTC device [e.g., /dev/async/syac@60(60000000)].

The Inet Addr field specifies the Internet address that will be assigned to the VTC. The Internet address is specified in dot format (see inet(3N)). By default, all ttys associated with this VTC will respond to telnet connections to this address. This behavior may be altered via tty entries in this file or by using the syac_ttyaddrs command.

The BAddr field specifies, in dot format (see inet(3N)), the broadcast address for the network to which the VTC is attached.

The Netmask field specifies, in dot format (see inet(3N)), the netmask for the network to which the VTC is attached.

The Route Info field specifies the location of the routing information that should be communicated to the VTC. The value of this field should be either the keyword default or the full pathname of a file containing routing information. If the keyword default is specified, then the routing information read from the host routing

table will be communicated to the VTC. If a pathname is specified, then the named file is read and the routing information in the file is communicated to the VTC. The format of the file should be identical to that of `/etc/gateways` (see `gateways`(4)).

It is an error if a configured SYAC VTC does not have a VTC specific entry in this file or if any of the fields are missing or blank.

Tty-specific Entries

The tty-specific entries are in the following format:
`<Tty path> <Inet Addr> [<Binary flag> [<Input flag>]]`

The `Tty path` specifies the full pathname of the tty device for the entry (eg `/dev/tty34`).

The `Inet Addr` specifies the Internet address to which the tty should respond to telnet connections. The Internet address should be in dot format (see `inet`(3N)).

The `Binary flag` field is an optional field which can have either the keyword `on` or the keyword `off` as its value. If the field is not present or if the keyword `on` is specified, then the VTC will attempt to negotiate telnet binary mode whenever a telnet connection is accepted for the tty in question. If the keyword `off` is specified, then the VTC will not attempt to negotiate telnet binary mode when a connection is accepted for the tty in question.

The `Input flag` field is an optional field which can have either the keyword `on` or the keyword `off` as its value. If the field is not present or if the keyword `on` is specified, then input processing will be unaffected by the state of telnet binary mode. If the keyword `off` is specified, then onboard input processing will be disabled when telnet binary mode is in effect on the line. Note that if this field is specified, then the `Binary flag` field must also be specified.

It is not necessary for each tty controlled by a VTC to have an entry in this file. By default, a tty will answer to connections to the Internet address of the VTC which controls the tty, telnet binary mode will be negotiated on when a connection is established, and input processing will be unaffected by the state of telnet binary mode. A tty should have an entry in this file only if this default behavior needs to be changed. The most common case would be to associate an Internet address with the tty that differs from the Internet address of the VTC that controls the tty.

Tty specific entries must be located after the VTC specific entry for their controlling VTC and before any other VTC specific entries.

EXAMPLE

```
/dev/async/syac@60(60000000) 128.222.3.112 128.222.3.255 255.255.255.0 default
/dev/tty34                   128.222.3.113
/dev/tty112                  128.222.3.84
/dev/async/syac@61(60020000) 128.222.3.96 128.222.3.255 255.255.255.0 /etc/syac1
/dev/tty260                  128.222.3.97  off          on
```

In this example, `syac@60` will have an Internet address of 128.222.3.112, a broadcast address of 128.222.3.255, and a netmask of 255.255.255.0. It will use the same routing information as the host computer. It should control `/dev/tty34` and `/dev/tty112`, each of which will answer telnet connections to a different Internet address than the VTC with which they are associated. The default behavior as regards telnet binary mode will apply to these two ttys and all other ttys controlled by syac@60.

Syac@61 will use the Internet address, broadcast address, and netmask specified above. The routing information in `/etc/syac1` will be communicated to the board

(the file should have the same format as `/etc/gateways`.) Syac@61 should control `/dev/tty260`, which will respond to telnet connections to the specified Internet address. For `/dev/tty260`, when a connection is established telnet binary mode will not be negotiated on, however, if the connection is negotiated into telnet binary mode (by a termserver, for example), on board input processing will be unaffected (this is the default).

FILES

`/etc/tcload/vtc.addrs` SYAC VTC configuration file

SEE ALSO

`syac_routes`(1M), `syac_ttyaddrs`(1M), `inet`(3N), `gateways`(4), `syac`(7), `termio`(7).

End of Chapter

Index

Note: Boldfaced page numbers (e.g., **1-5**) indicate definitions of terms or other key information.

E

F

G

H

I

K

L

M

N

O

P

R

S

Related Documents

The following list of related manuals gives titles of Data General manuals followed by nine-digit numbers used for ordering. You can order any of these manuals via mail or telephone (see the TIPS Order Form in the back of this manual).

For a complete list of AViiON® and DG/UX™ manuals, see the *Guide to AViiON® and DG/UX™ Documentation* (069-701085). The on-line version of this manual found in **/usr/release/doc_guide** contains the most current list.

Data General Software Manuals

User's Manuals

User's Reference for the DG/UX™ System
Contains an alphabetical listing of DG/UX, TCP/IP, and ONC/NFS manual pages for commands relating to general system operation. Ordering Number — 093-701054

Using the DG/UX™ Editors
Describes the text editors **vi** and **ed**, the batch editor **sed**, and the command line editor **editread**. Ordering Number — 069-701036

Using the DG/UX™ System
Describes the DG/UX system and its major features, including the C and Bourne shells, typical user commands, the file system, and communications facilities such as **mailx**. Ordering Number — 069-701035

Installation and Administration Manuals

Customizing the DG/UX™ System
Describes how to perform tasks that customize the DG/UX system to your site's needs. Included are descriptions of how to add user home directories, printers, terminals, third-party packages, operating system clients and secondary releases. Ordering Number — 093-701101

Installing the DG/UX™ System
Describes how to install the DG/UX operating system on AViiON hardware. Ordering Number — 093-701087

Managing the DG/UX™ System
Discusses the concepts and tasks related to DG/UX system management, providing general orientation to the administrator's job as well as instructions for managing disk resources, user profiles, files systems, printers and tape drives, and other features of the system. The manual approaches system administration through the **sysadm** facility. Ordering Number —

093-701088

Managing ONC™/NFS® and Its Facilities on the DG/UX™ System
Explains how to manage and use the DG/UX ONC™/NFS® product. Contains information on the Network File System (NFS), the Network Information Service (NIS), Remote Procedure Calls (RPC), and External Data Representation (XDR). Ordering Number — 093-701049

Managing TCP/IP on the DG/UX™ System
Explains how to prepare for the installation of Data General's TCP/IP (DG/UX) package on AViiON computer systems. Tells how to tailor the software for your site, use **sysadm** and **xsysadm** to manage the package and troubleshoot system problems. Ordering Number — 093-701051

Programming Manuals

Porting and Developing Applications on the DG/UX™ System
A compendium of useful information for experienced programmers developing or porting applications to the DG/UX™ system. It includes information on how to: set up your environment, use the software development tools, compile and link programs, port to the windowing environment, and build BCS applications. It also describes available debuggers and the various industry standards the DG/UX system supports. Ordering Number — 069-701059

Programmer's Reference for the DG/UX™ System, (Volume 1)
Alphabetical listing of manual pages for DG/UX programming commands and system calls. This is part of a three-volume set. Ordering Number — 093-701055

Programmer's Reference for the DG/UX™ System, (Volume 2)
Alphabetical listing of manual pages for DG/UX and ONC/NFS subroutines and libraries. This is part of a three-volume set. Ordering Number — 093-701056

Programmer's Reference for the DG/UX™ System, (Volume 3)
Alphabetical listing of manual pages for DG/UX, TCP/IP, and ONC/NFS file formats, miscellaneous features, and networking protocols. Part of a three-volume set, this volume contains the table of contents and index (**contents** (0) and **index** (0)) for man pages. Ordering Number — 093-701102

Writing a Standard Device Driver for the DG/UX™ System
Describes how to write a device driver for a DG/UX system running on an AViiON computer. Describes the drivers written to address specific devices or adapters that manage secondary bus access to specific devices. Information on kernel-level programming in the DG/UX system and descriptions of important kernel-level utility routines are found in *Programming in the DG/UX™ Kernel Environment* (093-701083). Ordering Number — 093-701053

End of Related Documents

TIPS ORDERING PROCEDURES

TO ORDER

1. An order can be placed with the TIPS group in two ways:
   a) MAIL ORDER – Use the order form on the opposite page and fill in all requested information. Be sure to include shipping charges and local sales tax. If applicable, write in your tax exempt number in the space provided on the order form.

      Send your order form with payment to:

      Data General Corporation
      ATTN: Educational Services/TIPS G155
      4400 Computer Drive
      Westboro, MA 01581–9973

   b) TELEPHONE – Call TIPS at (508) 870–1600 for all orders that will be charged by credit card or paid for by purchase orders over $50.00. Operators are available from 8:30 AM to 5:00 PM EST.

METHOD OF PAYMENT

2. As a customer, you have several payment options:
   a) Purchase Order – Minimum of $50. If ordering by mail, a hard copy of the purchase order must accompany order.
   b) Check or Money Order – Make payable to Data General Corporation.
   c) Credit Card – A minimum order of $20 is required for Mastercard or Visa orders.

SHIPPING

3. To determine the charge for UPS shipping and handling, check the total quantity of units in your order and refer to the following chart:

Total Quantity	Shipping & Handling Charge
1–4 Units	$5.00
5–10 Units	$8.00
11–40 Units	$10.00
41–200 Units	$30.00
Over 200 Units	$100.00

   If overnight or second day shipment is desired, this information should be indicated on the order form. A separate charge will be determined at time of shipment and added to your bill.

VOLUME DISCOUNTS

4. The TIPS discount schedule is based upon the total value of the order.

Order Amount	Discount
$1–$149.99	0%
$150–$499.99	10%
Over $500	20%

TERMS AND CONDITIONS

5. Read the TIPS terms and conditions on the reverse side of the order form carefully. These must be adhered to at all times.

DELIVERY

6. Allow at least two weeks for delivery.

RETURNS

7. Items ordered through the TIPS catalog may not be returned for credit.
8. Order discrepancies must be reported within 15 days of shipment date. Contact your TIPS Administrator at (508) 870–1600 to notify the TIPS department of any problems.

INTERNATIONAL ORDERS

9. Customers outside of the United States must obtain documentation from their local Data General Subsidiary or Representative. Any TIPS orders received by Data General U.S. Headquarters will be forwarded to the appropriate DG Subsidiary or Representative for processing.

TIPS ORDER FORM

Mail To: Data General Corporation
Attn: Educational Services/TIPS G155
4400 Computer Drive
Westboro, MA 01581 - 9973

BILL TO:

COMPANY NAME _____
ATTN: _____
ADDRESS _____
CITY _____
STATE _____ ZIP _____

SHIP TO: (No P.O. Boxes - Complete Only If Different Address)

COMPANY NAME _____
ATTN: _____
ADDRESS (NO PO BOXES) _____
CITY _____
STATE _____ ZIP _____

Priority Code _____ (See label on back of catalog)

Authorized Signature of Buyer (Agrees to terms & conditions on reverse side) | Title | Date | Phone (Area Code) | Ext.

ORDER #	QTY	DESCRIPTION	UNIT PRICE	TOTAL PRICE

A

☐ UPS

	ADD
1-4 Items	$ 5.00
5-10 Items	$ 8.00
11-40 Items	$ 10.00
41-200 Items	$ 30.00
200+ Items	$100.00

Check for faster delivery

Additional charge to be determined at time of shipment and added to your bill.

☐ UPS Blue Label (2 day shipping)
☐ Red Label (overnight shipping)

B VOLUME DISCOUNTS

Order Amount	Save
$0 – $149.99	0%
$150 – $499.99	10%
Over $500.00	20%

Tax Exempt # or Sales Tax (if applicable) _____

ORDER TOTAL	
Less Discount See B	–
SUB TOTAL	
Your local* sales tax	+
Shipping and handling – See A	+
TOTAL – See C	

C PAYMENT METHOD

☐ Purchase Order Attached ($50 minimum)
P.O. number is _____. (Include hardcopy P.O.)
☐ Check or Money Order Enclosed
☐ Visa ☐ MasterCard ($20 minimum on credit cards)

Account Number _____ Expiration Date _____

_____
Authorized Signature
(Credit card orders without signature and expiration date cannot be processed.)

THANK YOU FOR YOUR ORDER

PRICES SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.
PLEASE ALLOW 2 WEEKS FOR DELIVERY.
NO REFUNDS NO RETURNS.

* Data General is required by law to collect applicable sales or use tax on all purchases shipped to states where DG maintains a place of business, which covers all 50 states. Please include your local taxes when determining the total value of your order. If you are uncertain about the correct tax amount, please call 508–870–1600.

Form 702
Rev. 8/87

DATA GENERAL CORPORATION
TECHNICAL INFORMATION AND PUBLICATIONS SERVICE

TERMS AND CONDITIONS

Data General Corporation ("DGC") provides its Technical Information and Publications Service (TIPS) solely in accordance with the following terms and conditions and more specifically to the Customer signing the Educational Services TIPS Order Form. These terms and conditions apply to all orders, telephone, telex, or mail. By accepting these products the Customer accepts and agrees to be bound by these terms and conditions.

1. CUSTOMER CERTIFICATION
Customer hereby certifies that it is the owner or lessee of the DGC equipment and/or licensee/sub–licensee of the software which is the subject matter of the publication(s) ordered hereunder.

2. TAXES
Customer shall be responsible for all taxes, including taxes paid or payable by DGC for products or services supplied under this Agreement, exclusive of taxes based on DGC's net income, unless Customer provides written proof of exemption.

3. DATA AND PROPRIETARY RIGHTS
Portions of the publications and materials supplied under this Agreement are proprietary and will be so marked. Customer shall abide by such markings. DGC retains for itself exclusively all proprietary rights (including manufacturing rights) in and to all designs, engineering details and other data pertaining to the products described in such publication. Licensed software materials are provided pursuant to the terms and conditions of the Program License Agreement (PLA) between the Customer and DGC and such PLA is made a part of and incorporated into this Agreement by reference. A copyright notice on any data by itself does not constitute or evidence a publication or public disclosure.

4. LIMITED MEDIA WARRANTY
DGC warrants the CLI Macros media, provided by DGC to the Customer under this Agreement, against physical defects for a period of ninety (90) days from the date of shipment by DGC. DGC will replace defective media at no charge to you, provided it is returned postage prepaid to DGC within the ninety (90) day warranty period. This shall be your exclusive remedy and DGC's sole obligation and liability for defective media. This limited media warranty does not apply if the media has been damaged by accident, abuse or misuse.

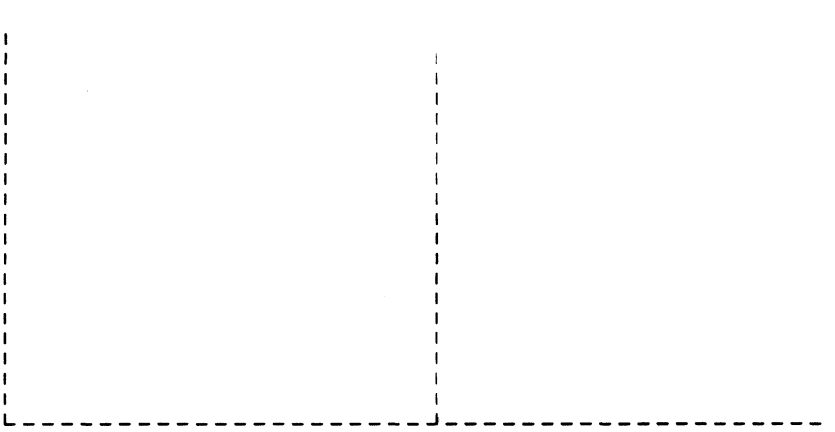
5. DISCLAIMER OF WARRANTY
EXCEPT FOR THE LIMITED MEDIA WARRANTY NOTED ABOVE, DGC MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE ON ANY OF THE PUBLICATIONS, CLI MACROS OR MATERIALS SUPPLIED HEREUNDER.

6. LIMITATION OF LIABILITY
A. CUSTOMER AGREES THAT DGC'S LIABILITY, IF ANY, FOR DAMAGES, INCLUDING BUT NOT LIMITED TO LIABILITY ARISING OUT OF CONTRACT, NEGLIGENCE, STRICT LIABILITY IN TORT OR WARRANTY SHALL NOT EXCEED THE CHARGES PAID BY CUSTOMER FOR THE PARTICULAR PUBLICATION OR CLI MACRO INVOLVED. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO CLAIMS FOR PERSONAL INJURY CAUSED SOLELY BY DGC'S NEGLIGENCE. OTHER THAN THE CHARGES REFERENCED HEREIN, IN NO EVENT SHALL DGC BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES WHATSOEVER, INCLUDING BUT NOT LIMITED TO LOST PROFITS AND DAMAGES RESULTING FROM LOSS OF USE, OR LOST DATA, OR DELIVERY DELAYS, EVEN IF DGC HAS BEEN ADVISED, KNEW OR SHOULD HAVE KNOWN OF THE POSSIBILITY THEREOF; OR FOR ANY CLAIM BY ANY THIRD PARTY.

B. ANY ACTION AGAINST DGC MUST BE COMMENCED WITHIN ONE (1) YEAR AFTER THE CAUSE OF ACTION ACCRUES.

7. GENERAL
A valid contract binding upon DGC will come into being only at the time of DGC's acceptance of the referenced Educational Services Order Form. Such contract is governed by the laws of the Commonwealth of Massachusetts, excluding its conflict of law rules. Such contract is not assignable. These terms and conditions constitute the entire agreement between the parties with respect to the subject matter hereof and supersedes all prior oral or written communications, agreements and understandings. These terms and conditions shall prevail notwithstanding any different, conflicting or additional terms and conditions which may appear on any order submitted by Customer. DGC hereby rejects all such different, conflicting, or additional terms.

8. IMPORTANT NOTICE REGARDING AOS/VS INTERNALS SERIES (ORDER #1865 & #1875)
Customer understands that information and material presented in the AOS/VS Internals Series documents may be specific to a particular revision of the product. Consequently user programs or systems based on this information and material may be revision–locked and may not function properly with prior or future revisions of the product. Therefore, Data General makes no representations as to the utility of this information and material beyond the current revision level which is the subject of the manual. Any use thereof by you or your company is at your own risk. Data General disclaims any liability arising from any such use and I and my company (Customer) hold Data General completely harmless therefrom.



Cut here and insert in binder spine pocket