AIX 5L™ Version 5.3

Commands Reference, Volume 5, s - u

SC23-4892-03

AIX 5L™ Version 5.3

Commands Reference, Volume 5, s - u

SC23-4892-03

Note
Before using this information and the product it supports, read the information in “Notices,” on page 647.

Fourth Edition (July 2006)

This edition applies to AIX 5L Version 5.3 and to all subsequent releases of this product until otherwise indicated in new editions.

A reader’s comment form is provided at the back of this publication. If the form has been removed, address comments to Information Development, Department 04XA-905-6C006, 11501 Burnet Road, Austin, Texas 78758-3493. To send comments electronically, use this commercial Internet address: aix6kpub@austin.ibm.com. Any information that you supply may be used without incurring any obligation to you.

© Copyright International Business Machines Corporation 1997, 2006. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About This Book ix
How to Use This Book ix
ISO 9000 xi
32-Bit and 64-Bit Support for the Single UNIX Specification xii
Related Information xii

Alphabetical Listing of Commands 1
sa Command 1
sa1 Command 3
sa2 Command 3
sact Command 4
sadc Command 5
sar Command 6
savebase Command 13
savecore Command 15
savevg Command 16
scan Command 18
sccs Command 20
sccsdiff Command 24
sccshelp Command 25
schedo Command 25
scls Command 43
script Command 44
sctpctrl Command 45
sdiff Command 46
secldapclntd Daemon 49
secldifconv Command 51
sectoldif Command 52
securetcpip Command 54
sed Command 55
sedmgr Command 59
send Command 63
sendbug Command 66
sendmail Command 66
setclock Command 72
setea Command 73
setgroups Command 74
setmaps Command 76
setsenv Command 79
settime Command 80
setuname Command 81
sh Command 82
shconf Command 83
shell Command 84
show Command 85
showmount Command 87
shutacct Command 89
shutdown Command 89
size Command 92
skulker Command 93
slattach Command 93
sleep Command 95
slibclean Command 95

© Copyright IBM Corp. 1997, 2006

sliplogin Command 96
slocal Command 100
smcaprop Command 100
smdefca Command 101
smdemon.cleanu Command 102
smexpcacert Command 103
smgenkeycr Command 104
smgenprivkr Command 104
smimpcacert Command 105
smimpservercert Command 106
sminstkey Command 107
smit Command 108
smitty Command 110
smlistcerts Command 112
smrsh Command 113
smserverprop Command 114
smsigncert Command 115
smtctl Command 115
smundefca Command 117
snap Command 118
snapcore Command 122
snapshot Command 124
snapsplit Command 125
snmpevent Command 127
snmpd Daemon 130
snmpdv1 Daemon 130
snmpdv3 Daemon 133
snmpinfo Command 136
snmpmibd Daemon 139
snmptrap Command 141
snmpv3_ssw Command 142
sno Command 143
sodebug Command 144
soelim Command 146
sort Command 147
sortbib Command 153
sortm Command 154
spell Command 156
spellin Command 158
spellout Command 158
split Command 159
splat Command 160
splitvg Command 169
splitlvcopy Command 170
splp Command 172
spost Command 174
spray Command 175
sprayd Daemon 177
srcmstr Daemon 178
startcondresp Command 179
start-secldapclntd Command 182
stop-secldapclntd Command 183
startrpdomain Command 184
startrpnode Command 187
startsrc Command 189
startup Command 190

startvsd Command . . . 191
startx Command . . . 192
statd Daemon . . . 195
statvsd Command . . . 195
stopcondresp Command . . . 197
stoprpdomain Command . . . 200
stoprpnode Command . . . 202
stopsrc Command . . . 204
stopvsd Command . . . 206
stpinet Method . . . 207
strace Command . . . 208
strchg Command . . . 209
strclean Command . . . 211
strconf Command . . . 211
strerr Daemon . . . 212
strinfo Command . . . 214
strings Command . . . 215
strip Command . . . 216
stripnm Command . . . 218
strload Command . . . 220
strreset Command . . . 223
strtune Command . . . 224
struct Command . . . 226
sttinet Method . . . 227
stty Command . . . 228
stty-cxma Command . . . 235
style Command . . . 237
su Command . . . 238
subj Command . . . 240
sum Command . . . 241
suma Command . . . 242
survd Daemon . . . 252
suspendvsd Command . . . 253
svmon Command . . . 255
swap Command . . . 280
swapoff Command . . . 281
swapon Command . . . 282
swcons Command . . . 284
swts Command . . . 285
sync Command . . . 286
synclvodm Command . . . 286
syncvg Command . . . 287
syscall Command . . . 289
sysck Command . . . 290
syscorepath Command . . . 293
sysdumpdev Command . . . 294
sysdumpstart Command . . . 298
sysline Command . . . 300
syslogd Daemon . . . 301
tab Command . . . 304
tabs Command . . . 305
tail Command . . . 309
talk Command . . . 311
talkd Daemon . . . 312
tapechk Command . . . 314
tar Command . . . 315

tbl Command . . . 321
tc Command . . . 324
tcbck Command . . . 325
tcopy Command . . . 330
tcpdump Command . . . 331
tctl Command . . . 341
tee Command . . . 343
telinit or init Command . . . 345
telnet, tn, or tn3270 Command . . . 348
telnetd Daemon . . . 359
termdef Command . . . 362
test Command . . . 363
tftp or utftp Command . . . 365
tftpd Daemon . . . 371
tic Command . . . 373
time Command . . . 374
timed Daemon . . . 375
timedc Command . . . 378
timex Command . . . 379
tip Command . . . 381
tokstat Command . . . 387
topas Command . . . 392
topasout Command . . . 404
topsvcs Command . . . 405
topsvcsctrl Command . . . 406
touch Command . . . 410
tprof Command . . . 412
tput Command . . . 422
tr Command . . . 423
trace Daemon . . . 427
traceroute Command . . . 433
tracesoff Command . . . 435
traceson Command . . . 437
trbsd Command . . . 438
trcctl Command . . . 440
trcdead Command . . . 441
trcevgrp Command . . . 442
trcnm Command . . . 444
trcrpt Command . . . 445
trcstop Command . . . 450
trcupdate Command . . . 451
troff Command . . . 452
trpt Command . . . 508
true or false Command . . . 512
truss Command . . . 513
tset Command . . . 517
tsh Command . . . 519
tsm Command . . . 521
tsort Command . . . 522
ttt Command . . . 523
tty Command . . . 524
tunchange Command . . . 526
tuncheck Command . . . 527
tundefault Command . . . 528
tunrestore Command . . . 529
tunsave Command . . . 531

turnacct Command 532
turnoff Command 533
turnon Command 533
tvi Command 534
twconvdict Command 537
twconvfont Command 538
type Command 539
ucfgif Method 540
ucfginet Method 540
ucfgqos Method 541
ucfgvsd Command 541
uconvdef Command 543
udefif Method 544
udefinet Method 544
udfcheck Command 545
udfcreate Command 546
udflabel Command 546
uil Command 547
uimx Command. 548
ul Command 549
ulimit Command 550
umask Command 552
umcode_latest Command 553
umount or unmount Command 555
umountall Command 556
unalias Command 558
uname Command 558
uncompress Command 561
undefvsd Command 562
unexpand Command 563
unfencevsd Command 564
unget Command (SCCS) 566
unifdef Command 567
uniq Command 568
units Command. 570
unlink Command 573
unloadipsec Command 574
unmirrorvg Command 574
unpack Command 576
untab Command 577
update Command 578
updatevsdnode Command 578
updatevsdtab Command 580
updatevsdvg Command 581
uprintfd Daemon 583
uptime Command 584
useradd Command 584
userdel Command. 587
usermod Command 588
users Command 590
usrck Command 590
utmpd Daemon 596
uucheck Command 597
uucico Daemon. 599
uuclean Command 601
uucleanup Command 602

uucp Command 604
uucpadm Command 608
uucpd Daemon 610
uudecode Command 611
uudemon.admin Command 612
uudemon.cleanu Command 613
uudemon.hour Command 615
uudemon.poll Command 616
uuencode Command 617
uuid_gen Command (NCS) 618
uukick Command 620
uulog Command 621
uuname Command 623
uupick Command 624
uupoll Command 626
uuq Command 627
uusched Daemon 629
uusend Command 630
uusnap Command 631
uustat Command 632
uuto Command 635
uutry Command 637
Uutry Command 638
uux Command 640
uuxqt Daemon 643

Appendix. Notices 647
Trademarks 648

Index 651

About This Book

This book provides end users with complete detailed information about commands for the AIX® operating system. The commands are listed alphabetically and by category, and complete descriptions are given for commands and their available flags. If applicable, each command listing contains examples. This volume contains AIX commands that begin with the letters s through u. This publication is also available on the documentation CD that is shipped with the operating system.

How to Use This Book

A command is a request to perform an operation or run a program. You use commands to tell the operating system what task you want it to perform. When commands are entered, they are deciphered by a command interpreter (also known as a shell) and that task is processed.

Some commands can be entered simply by typing one word. It is also possible to combine commands so that the output from one command becomes the input for another command. This is known as pipelining.

Flags further define the actions of commands. A flag is a modifier used with the command name on the command line, usually preceded by a dash.

Commands can also be grouped together and stored in a file. These are known as shell procedures or shell scripts. Instead of executing the commands individually, you execute the file that contains the commands.

Some commands can be constructed using Web-based System Manager applications or the System Management Interface Tool (SMIT).

Highlighting

The following highlighting conventions are used in this book:

Bold	Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.
Italics	Identifies parameters whose actual names or values are to be supplied by the user.
`Monospace`	Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

Format

Each command may include any of the following sections:

Purpose	A description of the major function of each command.
Syntax	A syntax statement showing command line options.
Description	A discussion of the command describing in detail its function and use.
Flags	A list of command line flags and associated variables with an explanation of how the flags modify the action of the command.
Parameters	A list of command line parameters and their descriptions.
Subcommands	A list of subcommands (for interactive commands) that explains their use.
Exit Status	A description of the exit values the command returns.
Security	Specifies any permissions needed to run the command.
Examples	Specific examples of how you can use the command.
Files	A list of files used by the command.
Related Information	A list of related commands in this book and related discussions in other books.

© Copyright IBM Corp. 1997, 2006

Reading Syntax Statements

Syntax statements are a way to represent command syntax and consist of symbols such as brackets ([ ]), braces ({ }), and vertical bars (|). The following is a sample of a syntax statement for the **unget** command:

unget [-**r***SID*] [-**s**] [-**n**] *File* ...

The following conventions are used in the command syntax statements:

- Items that must be entered literally on the command line are in **bold**. These items include the command name, flags, and literal charactors.
- Items representing variables that must be replaced by a name are in *italics*. These items include parameters that follow flags and parameters that the command reads, such as *Files* and *Directories*.
- Parameters enclosed in brackets are optional.
- Parameters enclosed in braces are required.
- Parameters not enclosed in either brackets or braces are required.
- A vertical bar signifies that you choose only one parameter. For example, [a | b] indicates that you *can* choose a, b, or nothing. Similarly, { a | b } indicates that you *must* choose either a or b.
- Ellipses (...) signify the parameter can be repeated on the command line.
- The dash (-) represents standard input.

Listing of Installable Software Packages

To list the installable software package (fileset) of an individual command use the **lslpp** command with the **-w** flag. For example, to list the fileset that owns the **installp** command, enter:

```
lslpp -w /usr/sbin/installp
```

Output similar to the following displays:

```
File                            Fileset                 Type
-----------------------------------------------------------------------
/usr/sbin/installp                bos.rte.install           File
```

To list the fileset that owns all file names that contain `installp`, enter:

```
lslpp -w "*installp*"
```

Output similar to the following displays:

```
File                            Fileset                 Type
-----------------------------------------------------------------------
/usr/sbin/installp                bos.rte.install           File
/usr/clvm/sbin/linstallpv         prpq.clvm                 File
/usr/lpp/bos.sysmgt/nim/methods/c_installp
                                  bos.sysmgt.nim.client     File
```

Running Commands in the Background

If you are going to run a command that takes a long time to process, you can specify that the command run in the background. Background processing is a useful way to run programs that process slowly. To run a command in the background, you use the & operator at the end of the command:

```
Command&
```

Once the process is running in the background, you can continue to work and enter other commands on your system.

At times, you might want to run a command at a specified time or on a specific date. Using the **cron** daemon, you can schedule commands to run automatically. Or, using the **at** and **batch** commands, you can run commands at a later time or when the system load level permits.

Entering Commands

You typically enter commands following the shell prompt on the command line. The shell prompt can vary. In the following examples, $ is the prompt.

To display a list of the contents of your current directory, you would type `ls` and press the Enter key:

```
$ ls
```

When you enter a command and it is running, the operating system does not display the shell prompt. When the command completes its action, the system displays the prompt again. This indicates that you can enter another command.

The general format for entering commands is:

Command Flag(s) Parameter

The flag alters the way a command works. Many commands have several flags. For example, if you type the **-l** (long) flag following the **ls** command, the system provides additional information about the contents of the current directory. The following example shows how to use the **-l** flag with the **ls** command:

```
$ ls -l
```

A parameter consists of a string of characters that follows a command or a flag. It specifies data, such as the name of a file or directory, or values. In the following example, the directory named **/usr/bin** is a parameter:

```
$ ls -l /usr/bin
```

When entering commands, it is important to remember the following:

- Commands are usually entered in lowercase.
- Flags are usually prefixed with a - (minus sign).
- More than one command can be typed on the command line if the commands are separated by a **;** (semicolon).
- Long sequences of commands can be continued on the next line by using the \ (backslash). The backslash is placed at the end of the first line. The following example shows the placement of the backslash:

  ```
  $ cat /usr/ust/mydir/mydata > \
  /usr/usts/yourdir/yourdata
  ```

When certain commands are entered, the shell prompt changes. Because some commands are actually programs (such as the **telnet** command), the prompt changes when you are operating within the command. Any command that you issue within a program is known as a subcommand. When you exit the program, the prompt returns to your shell prompt.

Operating system can operate with different shells (for example, Bourne, C, or Korn) and the commands that you enter are interpreted by the shell. Therefore, you must know what shell you are using so that you can enter the commands in the correct format.

Stopping Commands

If you enter a command and then decide to stop that command from running, you can halt the command from processing any further. To stop a command from processing, press the Interrupt key sequence (usually Ctrl-C or Alt-Pause). When the process is stopped, your shell prompt returns and you can then enter another command.

ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

32-Bit and 64-Bit Support for the Single UNIX Specification

Beginning with Version 5.2, the operating system is designed to support The Open Group's Single UNIX Specification Version 3 (UNIX 03) for portability of UNIX-based operating systems. Many new interfaces, and some current ones, have been added or enhanced to meet this specification, making Version 5.2 even more open and portable for applications, while remaining compatible with previous releases of AIX.
To determine the proper way to develop a UNIX 03-portable application, you may need to refer to The Open Group's UNIX 03 specification, which can be accessed online or downloaded from http://www.unix.org/ .

Related Information

The following books contain information about or related to commands:

- *AIX 5L Version 5.3 Commands Reference, Volume 1*
- *AIX 5L Version 5.3 Commands Reference, Volume 2*
- *AIX 5L Version 5.3 Commands Reference, Volume 3*
- *AIX 5L Version 5.3 Commands Reference, Volume 4*
- *AIX 5L Version 5.3 Commands Reference, Volume 5*
- *AIX 5L Version 5.3 Commands Reference, Volume 6*
- *AIX 5L Version 5.3 Files Reference*
- *Printers and printing*
- *Installation and migration*
- *AIX 5L Version 5.3 AIX Installation in a Partitioned Environment*
- *AIX 5L Version 5.3 Network Information Services (NIS and NIS+) Guide*
- *Performance management*
- *AIX 5L Version 5.3 Performance Tools Guide and Reference*
- *Security*
- *Networks and communication management*
- *Operating system and device management*
- *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 2*
- *AIX 5L Version 5.3 Technical Reference: Communications Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Communications Volume 2*
- *AIX 5L Version 5.3 Technical Reference: Kernel and Subsystems Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Kernel and Subsystems Volume 2*
- *AIX 5L Version 5.3 Web-based System Manager Administration Guide*
- *Performance Toolbox Version 2 and 3 for AIX: Guide and Reference*

Alphabetical Listing of Commands

sa Command

Purpose

Summarizes accounting records.

Syntax

/usr/sbin/sa [**-a**] [**-b**] [**-c**] [**-C**][**-d**] [**-D**] [**-i**] [**-j**] [**-k**] [**-K**] [**-l**] [**-m**] [**-n**] [**-r**] [**-s**] [**-t**] [**-u**] [**-v***Number* [**-f**]] [**-S***SaveFile*] [**-U***UserFile*] [*File* ...]

Description

The **sa** command summarizes the information in the file that collects the raw accounting data, either the **/var/adm/pacct** file or the file specified by the *File* parameter, and writes a usage summary report to the **/var/adm/savacct** file. Then, the **sa** command deletes the data in the **/var/adm/pacct** file so it can collect new accounting information. The next time the **sa** command executes, it reads the usage summary and the new data and incorporates all the information in its report.

The flags used with the **sa** command vary the type of information that is reported. The reports can contain the following fields:

`avio`	Indicates the average number of I/O operations per execution.
`cpu`	Indicates the sum of user and system time (in minutes).
`k`	Indicates the average K-blocks of CPU-time per execution.
`k*sec`	Indicates the CPU storage integral in kilo-core seconds.
`re`	Indicates the minutes of real time.
`s`	Indicates the minutes of system CPU time.
`tio`	Indicates the total number of I/O operations.
`u`	Indicates the minutes of user CPU time.

If you run the **sa** command without specifying any flags, the summary report includes the number of times each command was called as well as the `re`, `cpu`, `avio`, and `k` fields.

> **Note:** The **-b**, **-d**, **-D**, **-k**, **-K**, and **-n** flags determine how output is sorted. If you specify more than one of these flags on the command line, only the last one specified will take effect.

Summary files created under this release of the base operating system are saved in a format that supports large user IDs (8 characters or longer). Summary files created under previous releases may be in the old format that supports only user IDs of up to 7 characters. The **sa** command recognizes and supports both formats of the summary file. If you need to convert old format summary files to the new format, use the **-C** flag instead of the **-s** flag. You need to do this conversion only once. After converting you can use either the **-s** or the **-C** flag.

Flags

-a	Prints all command names, including those with unprintable characters. Commands that were used once are placed under the `other` category.
-b	Sorts output by the sum of user and system time divided by the number of calls. Otherwise, output is the sum of user and system time.
-c	Prints the time used by each command as a percentage of the time used by all the commands. This is in addition to the user, system and real time.

© Copyright IBM Corp. 1997, 2006

-C	Merges the accounting file into the summary file. If the summary file is in the old format, it is converted into the new format.
-d	Sorts the output by the average number of disk I/O operations.
-D	Sorts and prints the output by the total number of disk I/O operations.
-f	Does not force interactive threshold compression. This flag must be used with the **-v** flag.
-i	Reads only the raw data, not the summary file.
-j	Prints the number of seconds per call instead of the total minutes per category.
-k	Sorts the output by the average CPU time.
-K	Sorts and prints the output by the CPU-storage integral.
-l	Separates system and user time, instead of combining them.
-m	Prints the number of processes and the number of CPU minutes for each user.
-n	Sorts output by the number of calls.
-r	Reverses the order of the sort.
-s	Merges the accounting file into the summary file.
-S *SaveFile*	Uses the specified saved file as the command summary file, instead of the **/var/adm/savacct** file.
-t	Prints the ratio of real time to the sum of user and system time for each command.
-u	Suspends all other flags and prints the user's numeric ID and the command name for each command.
-U *UserFile*	Uses the specified file instead of the **/var/adm/usracct** file to accumulate the per-user statistics printed by the **-m** flag.
-v *Number*	Types the name of each command used the specified number times or fewer. When queried, if you type `y` (yes), the command is added to the `junk` category and appears in future summaries as part of that category.

Examples

1. To summarize accounting records for all the commands in the **/var/adm/pacct** file, enter:

   ```
   sa -a
   ```

   Commands used only once are placed under the `other` field.
2. To summarize accounting records by average CPU time, enter:

   ```
   sa -k
   ```

Files

/usr/sbin/sa	Contains the **sa** command.
/etc/sa	Contains the symbolic link to the **sa** command.
/var/adm/pacct	Contains raw accounting records.
/var/adm/savacct	Contains summary accounting records.
/var/adm/usracct	Contains summary accounting records by user.

Related Information

The **acctcms** command, **acctcom** command, **acctcon1** or **acctcon2** command, **acctmerg** command, **acctprc1**, **acctprc2**, or **accton** command, **fwtmp** command, **runacct** command.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in the *Operating system and device management*.

Setting up an accounting subsystem in the *Operating system and device management* describes the steps you must take to establish an accounting system.

See the Accounting commands in the *Operating system and device management* for a list of accounting commands that can be run automatically or entered from the keyboard.

sa1 Command

Purpose

Collects and stores binary data in the **/var/adm/sa/sa***dd* file.

Syntax

/usr/lib/sa/sa1 [*Interval Number*]

Description

The **sa1** command is a shell procedure variant of the **sadc** command and handles all of the flags and parameters of that command. The **sa1** command collects and stores binary data in the **/var/adm/sa/sa***dd* file*,* where *dd* is the day of the month. The *Interval* and *Number* parameters specify that the record should be written *Number* times at *Interval* seconds. If you do not specify these parameters, a single record is written. You must have permission to write in the **/var/adm/sa** directory to use this command.

The **sa1** command is designed to be started automatically by the **cron** command. If the **sa1** command is not run daily from the **cron** command, the **sar** command displays a message about the nonexistence of the **/usr/lib/sa/sa1** data file.

Examples

To create a daily record of **sar** activities, place the following entry in your adm **crontab** file:

```
0 8-17 * * 1-5 /usr/lib/sa/sa1 1200 3 &
```

Files

/var/adm/sa	Specifies the directory containing the daily data files.
/var/adm/sa/sa*dd*	Contains the daily data file, where the *dd* parameter is a number representing the day of the month.
/usr/lib/sa/sa1	Contains the **sa1** command.

Related Information

The **sadc** command, **sar** command, **sa2** command.

The **cron** daemon.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in the *Operating system and device management*.

Setting up an accounting subsystem in the *Operating system and device management* describes the steps you must take to establish an accounting system.

See the Accounting commands in the *Operating system and device management* for a list of accounting commands that can be run automatically or entered from the keyboard.

sa2 Command

Purpose

Writes a daily report in the **/var/adm/sa/sar***dd* file*.*

Syntax

/usr/lib/sa/sa2

Description

The **sa2** command is a variant shell procedure of the **sar** command, which writes a daily report in the **/var/adm/sa/sar***dd* file, where *dd* is the day of the month. The **sa2** command handles all of the flags and parameters of the **sar** command.

The **sa2** command is designed to be run automatically by the **cron** command and run concurrently with the **sa1** command.

Examples

To run the **sa2** command daily, place the following entry in the root **crontab** file:

```
5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 3600 -ubcwyaqvm &
```

This will generate a daily report called **/var/adm/sa/sar***dd*. It will also remove a report more than one week old.

Files

/var/adm/sa	Specifies the directory containing the daily data files.
/var/adm/sa/sar*dd*	Contains daily data file, where the *dd* parameter is a number representing the day of the month.
/usr/lib/sa/sa2	The path to the shell script of the **sa2** command.

Related Information

The **sa1** command, **sadc** command, **sar** command.

The **cron** daemon.

System accounting, Setting up an accounting subsystem in *Operating system and device management*.

Accounting commands in *Operating system and device management* lists accounting commands that can be run automatically or entered from the keyboard.

---

sact Command

Purpose

Displays current SCCS file-editing status.

Syntax

sact *File ...*

Description

The **sact** command reads Source Code Control System (SCCS) files and writes to standard output the contents, if any, of the p-file associated with the specified value of the *File* variable. The p-file is created by the **get -e** command. If a **-** (minus sign) is specified for the *File* value, the **sact** command reads standard input and interprets each line as the name of an SCCS file. If the *File* value is a directory, the **sact** command performs its actions on all SCCS files.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

To display the contents of a p-file, enter:

```
sact File
```

Files

/usr/bin/sact Contains the path to the SCCS **sact** command.

Related Information

The **delta** command, **get** command, **sccs** command, **unget** command.

The **sccsfile** file format.

List of SCCS Commands, Source Code Control System (SCCS) Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

sadc Command

Purpose

Provides a system data collector report.

Syntax

/usr/lib/sa/sadc [*Interval Number*] [*Outfile*]

/usr/lib/sa/sa1 [*Interval Number*]

/usr/lib/sa/sa2

Description

The **sadc** command, the data collector, samples system data a specified number of times (*Number*) at a specified interval measured in seconds (*Interval*). It writes in binary format to the specified outfile or to the standard output. When both *Interval* and *Number* are not specified, a dummy record, which is used at system startup to mark the time when the counter restarts from 0, will be written. The **sadc** command is intended to be used as a backend to the **sar** command.

The operating system contains a number of counters that are incremented as various system actions occur. The various system actions include:

- System Configuration Parameters
- System unit utilization counters
- Buffer usage counters
- Disk and tape I/O activity counters
- Tty device activity counters
- Switching and subroutine counters

- File access counters
- Queue activity counters
- Interprocess communication counters

Note: The **sadc** command reports only local activity.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

To write 10 records of one second intervals to the **/tmp/rpt** binary file, enter:

```
sadc 1 10 /tmp/rpt
```

Files

/var/adm/sa/sa*dd*	Contains the daily data file, *dd* represents the day of the month.
/var/adm/sa/sar*dd*	Contains the daily report file, *dd* represents the day of the month.
/tmp/rpt	Contains the binary file used for input by the **sar** command.
/tmp/sa.adrf1	Contains the address file.

Related Information

The **sar** command, **sa1** command, **sa2** command, **timex** command.

The **cron** daemon.

Setting up an accounting subsystem in the *Operating system and device management*.

In the *Operating system and device management*, System accounting describes system accounting and Accounting commands lists accounting commands that can be run automatically or entered from the keyboard.

---

sar Command

Purpose

Collects, reports, or saves system activity information.

Syntax

/usr/sbin/sar [{ **-A** [**-M**] | [**-a**] [**-b**] [**-c**] [**-d**][**-k**] [**-m**] [**-q**] [**-r**] [**-u**] [**-v**] [**-w**] [**-y**] [**-M**] }] [**-P** *ProcessorIdentifier***,** ... | **ALL**] [**-e***hh* [**:***mm* [**:***ss*]]] [**-X***File*] [**-f***File*] [**-i***Seconds*] [**-o***File*] [**-s***hh* [**:***mm* [**:***ss*]]] [*Interval* [*Number*]]

Description

The **sar** command writes to standard output the contents of selected cumulative activity counters in the operating system. The accounting system, based on the values in the *Number* and *Interval* parameters, writes information the specified number of times spaced at the specified intervals in seconds. The default sampling interval for the *Number* parameter is 1 second. The collected data can also be saved in the file specified by the **-o** *File* flag.

The **sar** command extracts and writes to standard output records previously saved in a file. This file can be either the one specified by the **-f** flag or, by default, the standard system activity daily data file, the **/var/adm/sa/sa***dd* file, where the *dd* parameter indicates the current day.

Without the **-P** flag, the **sar** command reports system-wide (global among all processors) statistics, which are calculated as averages for values expressed as percentages, and as sums otherwise. If the **-P** flag is given, the **sar** command reports activity which relates to the specified processor or processors. If **-P ALL** is given, the **sar** command reports statistics for each individual processor, followed by system-wide statistics.

You can select information about specific system activities using flags. Not specifying any flags selects only system unit activity. Specifying the **-A** flag selects all activities. The **sar** command prints the number of CPUs and the number of disks that are currently active before starting to print the statistics.

The default version of the **sar** command (CPU utilization report) might be one of the first facilities the user runs to begin system activity investigation, because it monitors major system resources. If CPU utilization is near 100 percent (user + system), the workload sampled is CPU-bound. If a considerable percentage of time is spent in I/O wait, it implies that CPU execution is blocked waiting for disk I/O. The I/O may be required file accesses or it may be I/O associated with paging due to a lack of sufficient memory.

Note: The time the system spends waiting for *remote* file access is *not* accumulated in the I/O wait time. If CPU utilization and I/O wait time for a task are relatively low, and the response time is not satisfactory, consider investigating how much time is being spent waiting for remote I/O. Since no high-level command provides statistics on remote I/O wait, trace data may be useful in observing this. If there is a change in system configuration that affects the output of the **sar** command, **sar** prints the average values up to the current iteration and then a warning message about the configuration change. It then continues the output, after printing the updated system configuration information.

Methods Used to Compute CPU Disk I/O Wait Time

AIX 4.3.3 and later contains enhancements to the method used to compute the percentage of CPU time spent waiting on disk I/O (*wio* time).The method used in AIX 4.3.2 and earlier versions of the operating system can, under certain circumstances, give an inflated view of wio time on SMPs. The wio time is reported by the commands **sar** (*%wio*), **vmstat** (*wa*) and **iostat** (*% iowait*).

The method used in AIX 4.3.2 and earlier versions is as follows: At each clock interrupt on each processor (100 times a second per processor), a determination is made as to which of the four categories (usr/sys/wio/idle) to place the last 10 ms of time. If the CPU was busy in usr mode at the time of the clock interrupt, then usr gets the clock tick added into its category. If the CPU was busy in kernel mode at the time of the clock interrupt, then the sys category gets the tick. If the CPU was not busy, a check is made to see if any I/O to disk is in progress. If any disk I/O is in progress, the wio category is incremented. If no disk I/O is in progress and the CPU is not busy, the idle category gets the tick. The inflated view of wio time results from all idle CPUs being categorized as wio regardless of the number of threads waiting on I/O. For example, systems with just one thread doing I/O could report over 90 percent wio time regardless of the number of CPUs it has.

The method used in operating system AIX 4.3.3 and later is as follows: The change in operating system AIX 4.3.3 is to only mark an idle CPU as wio if an outstanding I/O was started on that CPU. This method can report much lower wio times when just a few threads are doing I/O and the system is otherwise idle. For example, a system with four CPUs and one thread doing I/O will report a maximum of 25 percent wio time. A system with 12 CPUs and one thread doing I/O will report a maximum of 8 percent wio time. NFS client reads/writes go through the VMM, and the time that biods spend in the VMM waiting for an I/O to complete is now reported as I/O wait time.

If multiple samples and multiple reports are desired, it is convenient to specify an output file for the **sar** command. Direct the standard output data from the **sar** command to `/dev/null` and run the **sar** command as a background process. The syntax for this is:

```
sar -A -o data.file interval count > /dev/null &
```

All data is captured in binary form and saved to a file (`data.file`). The data can then be selectively displayed with the **sar** command using the **-f** option.

The **sar** command calls a process named **sadc** to access system data. Two shell scripts (`/usr/lib/sa/sa1` and `/usr/lib/sa/sa2`) are structured to be run by the **cron** command and provide daily statistics and reports. Sample stanzas are included (but commented out) in the **/var/spool/cron/crontabs/adm** crontab file to specify when the **cron** daemon should run the shell scripts. Collection of data in this manner is useful to characterize system usage over a period of time and determine peak usage hours.

You can insert a dummy record into the standard system activity daily data file at the time of system start by un-commenting corresponding lines in the **/etc/rc** script. The **sar** command reports `time change not positive` for any record where CPU times are less than the previous record. This occurs if you reboot the system with the dummy record insertion lines in **/etc/rc** commented out.

Beginning with AIX 5.3, the **sar** command reports utilization metrics `physc` and `%entc` which are related to Micro-Partitioning and simultaneous multi-threading environments. These metrics will only be displayed on Micro-Partitioning and simultaneous multi-threading environments. `physc` indicates the number of physical processors consumed by the partition (in case of system wide utilization) or logical CPU (if the **-P** flag is specified) and `%entc` indicates the percentage of the allocated entitled capacity (in case of system wide utilization) or granted entitled capacity (if the **-P** flag is specified). When the partition runs in capped mode, the partition cannot get more capacity than it is allocated. In uncapped mode, the partition can get more capacity than it is actually allocated. This is called granted entitled capacity. If the **-P** flag is specified and there is unused capacity, **sar** prints the unused capacity as separate CPU with `cpu id` U.

Note: The **sar** command only reports on local activities.

You can use the System application in Web-based System Manager (wsm) to run this command.

You could also use the System Management Interface Tool (SMIT) **smit sar** fast path to run this command.

Flags

-A Without the **-P** flag, using the **-A** flag is equivalent to specifying **-abcdkmqruvwy**. When used with the **-P** flag, the **-A** is equivalent to specifying **-acmuw**. Without the **-M** flag, headers are only printed once in multiple lines grouped together before the data for the first interval. When this flag is used with the **-M** flag, each line of data at each iteration is preceded by the appropriate header.

-a Reports use of file access system routines specifying how many times per second several of the system file access routines have been called. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed:

dirblk/s
Number of 512-byte blocks read by the directory search routine to locate a directory entry for a specific file.

iget/s Calls to any of several i-node lookup routines that support multiple file system types. The **iget** routines return a pointer to the i-node structure of a file or device.

lookuppn/s
Calls to the directory search routine that finds the address of a v-node given a path name.

-b Reports buffer activity for transfers, accesses, and cache (kernel block buffer cache) hit ratios per second. Access to most files in Version 3 bypasses kernel block buffering and therefore does not generate these statistics. However, if a program opens a block device or a raw character device for I/O, traditional access mechanisms are used making the generated statistics meaningful. The following values are displayed:

bread/s, bwrit/s
Reports the number of block I/O operations. These I/Os are generally performed by the kernel to manage the block buffer cache area, as discussed in the description of the **lread/s** value.

lread/s, lwrit/s
Reports the number of logical I/O requests. When a logical read or write to a block device is performed, a logical transfer size of less than a full block size may be requested. The system accesses the physical device units of complete blocks and buffers these blocks in the kernel buffers that have been set aside for this purpose (the block I/O cache area). This cache area is managed by the kernel, so that multiple logical reads and writes to the block device can access previously buffered data from the cache and require no real I/O to the device. Application read and write requests to the block device are reported statistically as logical reads and writes. The block I/O performed by the kernel to the block device in management of the cache area is reported as block reads and block writes.

pread/s, pwrit/s
Reports the number of I/O operations on raw devices. Requested I/O to raw character devices is not buffered as it is for block devices. The I/O is performed to the device directly.

%rcache, %wcache
Reports caching effectiveness (cache hit percentage). This percentage is calculated as: [(100)x(lreads - breads)/ (lreads)].

-c Reports system calls. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed:

exec/s, fork/s
Reports the total number of **fork** and **exec** system calls.

sread/s, swrit/s
Reports the total number of read/write system calls.

rchar/s, wchar/s
Reports the total number of characters transferred by read/write system calls.

scall/s Reports the total number of system calls.

Note: The **sar** command itself can generate a considerable number of reads and writes depending on the interval at which it is run. Run the **sar** statistics without the workload to understand the **sar** command's contribution to your total statistics.

-d Reports activity for each block device with the exception of tape drives. The activity data reported is:

%busy Reports the portion of time the device was busy servicing a transfer request.

avque Before AIX 5.3: Reports the instantaneous number of requests sent to disk but not completed yet. AIX 5.3: Reports the average number of requests waiting to be sent to disk.

read/s, write/s, blk/s
Reports the read-write transfers from or to a device in kilobytes/second.

avwait, avserv
Average wait time and service time per request in milliseconds.

-e *hh*[:*mm*[:*ss*]] Sets the ending time of the report. The default ending time is 18:00.

-X *File* Extracts the records from the *File*, which is generated by AIX 4.3 or AIX 4.2 version of the **sar/sadc** command.

-f *File* Extracts records from the *File* (created by **-o** *File* flag). The default value of the *File* parameter is the current daily data file, the **/var/adm/sa/sa***dd* file.

-i *Seconds* Selects data records at seconds as close as possible to the number specified by the *Seconds* parameter. Otherwise, the **sar** command reports all seconds found in the data file.

-k Reports kernel process activity. The following values are displayed:

kexit/s Reports the number of kernel processes terminating per second.

kproc-ov/s
Reports the number of times kernel processes could not be created because of enforcement of process threshold limit.

ksched/s
Reports the number of kernel processes assigned to tasks per second.

-M Enables multiple headers in output when used with at least two combinations of [**abckmqruvwy**] or with the **-A** flag. In this mode, each line of data is preceded by the corresponding header at each iteration. This flag is ignored when used without [*Interval* [*Number*]].

-m Reports message (sending and receiving) and semaphore (creating, using, or destroying) activities per second. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following values are displayed:

msg/s Reports the number of IPC message primitives.

sema/s
Reports the number of IPC semaphore primitives.

-o *File* Saves the readings in the file in binary form. Each reading is in a separate record and each record contains a tag identifying the time of the reading.

-P *ProcessorIdentifier***,** ... | **ALL** Reports per-processor statistics for the specified processor or processors. Specifying the **ALL** keyword reports statistics for each individual processor, and globally for all processors . Of the flags which specify the statistics to be reported, only the **-a**, **-c**, **-m**, **-u**, and **-w** flags are meaningful with the **-P** flag.

-q Reports queue statistics. The following values are displayed:

runq-sz
Reports the average number of kernel threads in the run queue.

%runocc
Reports the percentage of the time the run queue is occupied.

swpq-sz
Reports the average number of kernel threads waiting to be paged in.

%swpocc
Reports the percentage of the time the swap queue is occupied.

Note: A blank value in any column indicates that the associated queue is empty.

-r Reports paging statistics. The following values are displayed:

cycle/s Reports the number of page replacement cycles per second.

fault/s Reports the number of page faults per second. This is not a count of page faults that generate I/O, because some page faults can be resolved without I/O.

slots Reports the number of free pages on the paging spaces.

odio/s Reports the number of non paging disk I/Os per second.

-s *hh*[:*mm*[:*ss*]]
Sets the starting time of the data, causing the **sar** command to extract records time-tagged at, or following, the time specified. The default starting time is 08:00.

-u
Reports per processor or system-wide statistics. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. Because the **-u** flag information is expressed as percentages, the system-wide information is simply the average of each individual processor's statistics. Also, the I/O wait state is defined system-wide and not per processor. The following values are displayed:

%idle Reports the percentage of time the cpu or cpus were idle with no outstanding disk I/O requests.

%sys Reports the percentage of time the cpu or cpus spent in execution at the system (or kernel) level.

%usr Reports the percentage of time the cpu or cpus spent in execution at the user (or application) level.

%wio Reports the percentage of time the cpu(s) were idle during which the system had outstanding disk/NFS I/O request(s). See detailed description above.

physc Reports the number of physical processors consumed. This will be reported only if the partition is running with shared processors or simultaneous multi-threading enabled.

%entc Reports the percentage of entitled capacity consumed. This will be reported only if the partition is running with shared processors. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

Note: The **sar** command reports system unit activity if no other specific content options are requested. If the **-P** flag is used and the partition is running with shared processors, and if the partition capacity usage is what is allocated, then a CPU row with `cpuid U` will be reported to show the system-wide unused capacity. If the partition is running with shared processors in uncapped mode, then `%entc` will report the percentage of granted entitled capacity against each CPU row and percentage of allocated entitled capacity in the system-wide CPU row.

-v
Reports status of the process, kernel-thread, i-node, and file tables. The following values are displayed:

file-sz, inod-sz, proc-sz , thrd-sz
Reports the number of entries in use for each table.

-w
Reports system switching activity. When used with the **-P** flag, the information is provided for each specified processor; otherwise, it is provided only system-wide. The following value is displayed:

pswch/s
Reports the number of context switches per second.

-y Reports tty device activity per second.

canch/s
Reports tty canonical input queue characters. This field is always 0 (zero) for AIX Version 4 and later versions.

mdmin/s
Reports tty modem interrupts.

outch/s
Reports tty output queue characters.

rawch/s
Reports tty input queue characters.

revin/s Reports tty receive interrupts.

xmtin/s
Reports tty transmit interrupts.

Security

Access Control: These commands should grant execute (x) access only to members of the **adm** group.

Examples

1. To report system unit activity, enter:

   ```
   sar
   ```

2. To report current tty activity for each 2 seconds for the next 40 seconds, enter:

   ```
   sar  -y  -r 2 20
   ```

3. To watch system unit for 10 minutes and sort data, enter:

   ```
   sar  -o temp 60 10
   ```

4. To report cpu activity for the first two processors, enter:

   ```
   sar  -u  -P 0,1
   ```

   This produces output similar to the following:

   ```
   cpu  %usr  %sys  %wio  %idle
   0      45    45     5      5
   1      27    65     3      5
   ```

5. To report message, semaphore, and cpu activity for all processors and system-wide, enter:

   ```
   sar  -mu  -P ALL
   ```

   On a four-processor system, this produces output similar to the following (the last line indicates system-wide statistics for all processors):

   ```
   cpu  msgs/s  sema/s  %usr  %sys  %wio  %idle
   0      7       2       45    45     5     5
   1      5       0       27    65     3     5
   2      3       0       55    40     1     4
   3      4       1       48    41     4     7
   -     19       3       44    48     3     5
   ```

6. To see physical processor consumed and entitlement consumed for all processors system-wide, run **sar** command in a shared processor logical partition machine, as follows:

   ```
   sar –P ALL
   ```

On a two-logical processor system, this produces output similar to the following (the last line indicates system-wide statistics for all processors, and the line with `cpuid U` indicates the system-wide Unused capacity):

```
cpu    %usr    %sys    %wio   %idle   physc   %entc
 0        0       0       0     100    0.02     3.1
 1        0       0       0     100    0.00     1.0
 U        -       -       0      96    0.48    96.0
 -        0       0       0     100    0.02     4.0
```

7. To report system call, kernel process, and paging activities with separate headers for each of the three lines of data at each iteration for every 2 seconds for the next 40 seconds, enter:

   ```
   sar -Mckr 2 20
   ```

8. To report all activities with multiple sets of headers for every 2 seconds for the next 40 seconds, enter:

   ```
   sar -MA 2 20
   ```

Files

/usr/sbin/sar	Contains the **sar** command.
/bin/sar	Indicates the symbolic link to the **sar** command.
/var/adm/sa/sa*dd*	Indicates the daily data file, where the *dd* parameter is a number representing the day of the month.

Related Information

The **sadc** command, **sa1** command, **sa2** command.

System accounting in the *Operating system and device management*.

Setting up an accounting subsystem in the *Operating system and device management*.

Accounting commands in the *Operating system and device management* lists accounting commands that can be run automatically or entered from the keyboard.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

savebase Command

Purpose

Saves information about base-customized devices in the Device Configuration database onto the boot device.

Syntax

savebase [**-o** *Path*] [**-d** *File*] [**-v**]

Description

The **savebase** command stores customized information for base devices for use during phase 1 of system boot. By default, the **savebase** command retrieves this information from the **/etc/objrepos** directory. However, you can override this action by using the **-o** flag to specify an ODM directory. The **savebase** command is typically run without any parameters. It uses the **/dev/ipl_blv** special file link to identify the output destination.

Alternatively, use the **-d** flag to specify a destination file or a device, such as the **/dev/hdisk0** device file. To identify a specific output destination, the **-d** flag identifies the file to which **savebase** writes the base

customized device data. This file can be either a regular file or a device special file. The device special file identifies either a disk device special file or a boot logical volume device special file.

A disk device special file can be used where there is only one boot logical volume on the disk. The **savebase** command ensures that the given disk has only one boot logical volume present and is bootable. If neither of these conditions is true, **savebase** does not save the base customized device data to the disk and exits with an error.

When a second boot logical volume is on a disk, the boot logical volume device special file must be used as the destination device to identify which boot image the base customized device data will be stored in. A boot logical volume device special file can be used even if there is only one boot logical volume on the disk. The **savebase** command ensures that the given device special file is a boot logical volume and it is bootable before saving any data to it. If either of these checks fails, **savebase** exits with an error.

The **savebase** command determines what device information to save using the `PdDv.base` field corresponding to each entry in the **CuDv** object class. Specifically, the `PdDv.base` field is a bit mask which represents the type of boot for which this device is a base device. The **savebase** command determines the current type of boot by accessing the **boot_mask** attribute in the **CuAt** object class. The value of this attribute is the bit mask to apply to the `PdDv.base` field to determine which devices are base.

Note: Base devices are those devices that get configured during phase 1 boot; they may vary depending on the type of boot (mask). For example, if the mask is **NETWORK_BOOT**, network devices are considered base; for **DISK_BOOT**, disk devices are considered base. The type-of-boot masks are defined in the **/usr/include/sys/cfgdb.h** file.

Note: The **-m** flag is no longer used by the **savebase** command. For compatibility reasons, the flag can be specified, but **savebase** effectively ignores it.

Flags

-d *File* Specifies the destination file or device to which the base information will be written.
-o *Path* Specifies a directory containing the Device Configuration database.
-v Causes verbose output to be written to standard input.

Examples

1. To save the base customized information and see verbose output, enter:
   ```
   savebase -v
   ```
2. To specify an ODM directory other than the **/usr/lib/objrepos** directory, enter:
   ```
   savebase -o /tmp/objrepos
   ```
3. To save the base customized information to the **/dev/hdisk0** device file instead of to the boot disk, enter:
   ```
   savebase -d /dev/hdisk0
   ```

Files

/usr/include/sys/cfgdb.h Defines the type of boot mask for base devices.
/usr/lib/objrepos/PdDv Contains entries for all known device types supported by the system.
/etc/objrepos/CuDv Contains entries for all device instances defined in the system.
/etc/objrepos/CuAt Contains customized device-specific attribute information.
/etc/objrepos/CuDep Describes device instances that depend on other device instances.
/etc/objrepos/CuDvDr Stores information about critical resources that need concurrency management through the use of the Device Configuration Library routines.

Related Information

The **bosboot** command, **restbase** command.

Object Data Manager (ODM) Overview for Programmers in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

Device Configuration Subsystem: Programming Introduction, List of Device Configuration Commands in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

---

savecore Command

Purpose

Saves a system dump.

Syntax

savecore { [[**-c**] [**-d**] [**-f**]] | [**-F** [**-d**]] } *DirectoryName SystemName*

Description

The function of the **savecore** command is to save a system dump and is usually run at system startup.

The **savecore** command checks to see that you have a recent dump and that there is enough space to save it. The system dump is saved in the *DirectoryName*/**vmcore.***n* file, and the system is saved in the *DirectoryName*/**vmunix.***n* file. The *n* variable is specified in the *DirectoryName*/**bounds** file. If this file does not exist, it is created with a default of **0**, and the *n* variable uses this value. With each subsequent dump, the *n* variable is increased by 1.

The **savecore** command also checks to see if the current dump was compressed. If so, then it is copied to a file named *DirectoryName*/**vmcore**. *n*.**Z**, where .**Z** is the standard indication that a file is compressed.

> **Note**: This applies to AIX 4.3.2 and later.

If the system dump was from a system other than */unix*, the name of the system must be supplied as *SystemName*.

> **Note:** The **savecore** command saves only the current dump and the dump prior to the current one.

The directory may contain a file named **minfree**. This file contains the number of kbytes to leave free in the directory. The **minfree** file can be used to ensure a minimum amount of free space is left after the dump is copied.

Flags

-c Marks the dump invalid (not recent), but does not copy it.
-d Copies only the dump. It does not copy the system.
-f Copies the dump even if it appears to be invalid.
-F Reports the amount of space available for a dump in the copy directory. This may be more than the free space since the **savecore** command keeps the current dump and the previous dump, deleting others. No copying is done if the **-F** flag is specified. This flag is only valid with the **-d** flag.

Examples

1. To copy the dump (not the system) to *DirectoryName*, type:

```
savecore -d DirectoryName
```

2. To copy the dump even if it is invalid, type:
   ```
   savecore -f -d DirectoryName
   ```
3. To mark the dump invalid, type:
   ```
   savecore -c
   ```
4. To copy the dump and the system, type:
   ```
   savecore -d DirectoryName SystemName
   ```
5. To see how much space is available for a dump, type:
   ```
   savecore -d -F DirectoryName
   ```

Related Information

The **sysdumpdev** command, **sysdumpstart** command.

---

savevg Command

Purpose

Finds and backs up all files belonging to a specified volume group.

Syntax

savevg [**-a**] [**-A**] [**-b** *Blocks*] [**-e**] [**-f** *Device*] [**-i** | **-m**] [**-p**] [**-r**] [**-v**] [**-V**] [**-X**] *VGName*

Description

The **savevg** command finds and backs up all files belonging to a specified volume group. The volume group must be varied-on, and the file systems must be mounted. The **savevg** command uses the data file created by the **mkvgdata** command. This data file can be one of the following:

/image.data
: Contains information about the root volume group (**rootvg**). The **savevg** command uses this file to create a backup image that can be used by Network Installation Management (NIM) to reinstall the volume group to the current system or to a new system.

/tmp/vgdata/*vgname***/***vgname***.data**
: Contains information about a user volume group. The *VGName* variable reflects the name of the volume group. The **savevg** command uses this file to create a backup image that can be used by the **restvg** command to remake the user volume group.

To create a backup of the operating system to CD, use the **mkcd** command.

Note: The **savevg** command will not generate a bootable tape if the volume group is the root volume group. Although the tape is not bootable, the first three images on the tape are dummy replacements for the images normally found on a bootable tape. The actual system backup is the fourth image.

Flags

-a Does not backup extended attributes or NFS4 ACLs.

-A Backs up DMAPI file system files.

-b *Blocks* Specifies the number of 512-byte blocks to write in a single output operation. If this parameter is not specified, the **backup** command uses a default value appropriate for the physical device selected. Larger values result in larger physical transfers to tape devices. The value specified must be a multiple of the physical block size of the device being used.

-e Excludes files specified in the **/etc/exclude.***vgname* file from being backed up by this command.

Note: If you want to exclude certain files from the backup, create the **/etc/exclude.rootvg** file, with an ASCII editor, and enter the patterns of file names that you do not want included in your system backup image. The patterns in this file are input to the pattern matching conventions of the **grep** command to determine which files will be excluded from the backup. If you want to exclude files listed in the **/etc/exclude.rootvg** file, select the Exclude Files field and press the Tab key once to change the default value to yes.

For example, to exclude all the contents of the directory called scratch, edit the exclude file to read as follows:

```
/scratch/
```

For example, to exclude the contents of the directory called **/tmp**, and avoid excluding any other directories that have **/tmp** in the pathname, edit the exclude file to read as follows:

```
^./tmp/
```

All files are backed up relative to . (current working directory). To exclude any file or directory for which it is important to have the search match the string at the beginning of the line, use ^ (caret character) as the first character in the search string, followed by **.** (dot character), followed by the filename or directory to be excluded.

If the filename or directory being excluded is a substring of another filename or directory, use **^.** (caret character followed by dot character) to indicate that the search should begin at the beginning of the line and/or use **$** (dollar sign character) to indicate that the search should end at the end of the line.

-f *Device* Specifies the device or file name on which the image is to be stored. The default is the **/dev/rmt0** device.

-i Creates the data file by calling the **mkvgdata** command.

-m Creates the data file with map files by calling the **mkvgdata** command with the **-m** flag.

-p Disables software packing of the files as they are backed up. Some tape drives use their own packing or compression algorithms.

-r Backs up user volume group information and administration data files. This backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files if any exist. This does not backup user data files. This backup can be used to create a user volume group without restoring user data files. This cannot be done to rootvg.

-v Verbose mode. Lists files as they are backed up.

-V Verifies a tape backup. This flag causes **savevg** to verify the file header of each file on the backup tape and report any read errors as they occur.

-X Specifies to automatically expand the **/tmp** file system if necessary. The **/tmp** file system may need to be extended to make room for the boot image when creating a bootable backup to tape.

Parameters

VGName Specifies the name of the volume group to be backed up.

SMIT Fast Paths

1. To list the contents of a root volume group backup that is created with the **savevg** command, enter the following SMIT fast path:

   ```
   smit lsmksysb
   ```

2. To list the contents of a user volume group backup that is created with the **savevg** command, enter the following SMIT fast path:

   ```
   smit lsbackvg
   ```

3. To restore individual files from a root volume group backup, enter the following SMIT fast path:

```
smit restmksysb
```

4. To restore individual files from a user volume group backup, enter the following SMIT fast path:

```
smit restsavevg
```

Examples

1. To backup the root volume group (operating system image) to the **/mysys/myvg/myroot** backup file and create an **/image.data** file, enter:

```
savevg -i -f/mysys/myvg/myroot rootvg
```

2. To backup the **uservg** volume group to the default tape drive (**dev/rmt0**) and create a new **uservg.data** file, enter:

```
savevg -i uservg
```

3. To backup the **data2** volume group and create map files along with a new **data2.data** file on **rmt1** device, enter:

```
savevg -mf/dev/rmt1 data2
```

4. To backup the **data2** volume group, excluding the files listed in the **/etc/exclude.data2** file, enter:

```
savevg -ief/dev/rmt1 data2
```

5. To backup the volume group **my_vg** to the tape in **/dev/rmt0** and then verify the readability of file headers, enter:

```
savevg -f /dev/rmt0 -V my_vg
```

Files

/image.data	Used when the volume group is **rootvg**.
/tmp/vgdata/*vgname* **/***vgname***.data**	Used when the volume group is not **rootvg** and where *vgname* is the name of the volume group.

Related Information

The **backup** command, **bosboot** command, **mkcd** command, **mkszfile** command.

---

scan Command

Purpose

Produces a one line per message scan listing.

Syntax

scan [**+***Folder*] [*Messages*] [**-form** *FormFile* | **-format** *String*] [**-noheader** | **-header**] [**-clear** | **-noclear**] [**-help**]

Description

The **scan** command displays a line of information about the messages in a specified folder. Each line gives the message number, date, sender, subject, and as much of the message body as possible. By default, the **scan** command displays information about all of the messages in the current folder.

If a + (plus sign) is displayed after the message number, the message is the current message in the folder. If a - (minus sign) is displayed, you have replied to the message. If an * (asterisk) is displayed after the date, the `Date:` field was not present and the displayed date is the last date the message was changed.

Flags

-clear Clears the display after sending output. The **scan** command uses the values of the **$TERM** environment variable to determine how to clear the display. If standard output is not a display, the **scan** command sends a form feed character after sending the output.

+*Folder* Specifies which folder to scan. The default is the current folder.

-form *FormFile* Displays the **scan** command output in the alternate format described by the *FormFile* variable.

-format *String* Displays the **scan** command output in the alternate format described by the *String* variable.

-header Displays a heading that lists the folder name and the current date and time.

-help Lists the command syntax, available switches (toggles), and version information.

Note: For Message Handler (MH), the name of this flag must be fully spelled out.

Messages Displays information about each specified message in the specified folder. You can use the following references when specifying messages:

Number
Specifies the number of the message.

Sequence
Specifies a group of messages specified by the user. Recognized values include:

all All messages in a folder. This is the default.

cur or . (period)
Current® message.

first First message in a folder.

last Last message in a folder.

next Message following the current message.

prev Message preceding the current message.

-noclear Prevents clearing of the terminal after sending output. This is the default.

-noheader Prevents display of a heading. This is the default.

-width *Number* Sets the number of columns in the **scan** command output. The default is the width of the display.

Profile Entries

The following entries are entered in the *UserMhDirectory*/**.mh_profile** file:

`Alternate-Mailboxes:` Specifies the mailboxes.

`Current-Folder:` Sets the default current folder.

`Path:` Specifies the *UserMhDirectory*.

Examples

1. To get a one-line list of all the messages in the current folder, enter:

   ```
   scan
   ```

   The system responds with a message similar to the following:

   ```
   3   04/17 dale@athena Status meeting <<The weekly status meeting
   5   04/20 tom@venus   Due Dates      <<Your project is due to
   6   04/21 dawn@tech   Writing Clas   <<There will be a writing
   ```

2. To get a one-line list of messages `11` through `15` in the `test` folder, enter:

```
scan  +test 11-15
```

The system responds with a message similar to the following:

```
11  04/16 karen@anchor Meeting    <<Today's meeting is at 2 p.m.
12  04/18 tom@venus    Luncheon   <<There will be a luncheon to
14  04/20 dale@athena  First Draft  <<First drafts are due
15  04/21 geo@gtwn     Examples   <<The examples will be written
```

Files

$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/scan.size	Contains a sample scan format string.
/etc/mh/scan.time	Contains a sample scan format string.
/etc/mh/scan.timely	Contains a sample scan format string.
/usr/bin/scan	Contains the executable form of the **scan** command.

Related Information

The **inc** command, **pick** command, **show** command.

The **.mh_alias** file format, **.mh_profile** file format.

Mail applications in *Networks and communication management*.

---

sccs Command

Purpose

Administration program for SCCS commands.

Syntax

sccs [**-r**] [**-d***Path*] [**-p***Path*] *Command* [*CommandFlags*] *File ...*

Description

The **sccs** command is an administration program that incorporates the set of Source Code Control System (SCCS) commands into the operating system. Additionally, the **sccs** command can be used to assign or reassign file ownership (see the **-r** flag).

The **sccs** command activates a specified *Command* having the specified flags and arguments. Each file is normally placed in a directory named SCCS and named **s.filename**. The directory SCCS is assumed to exist relative to the working directory (unless the **-p** flag is used).

Two types of commands can be used in the **sccs** command syntax sentence. The first type consists of 14 **sccs** commands that can be entered at the prompt. The second type, pseudo-commands, can be used only as part of the **sccs** command syntax. There are 12 pseudo-commands, which perform the following actions:

edit	Equivalent to the **get -e** command.

delget Performs a **delta** command on the named files and then gets new versions. The new versions of the files have expanded identification keywords and are not editable.

Flags:

-m, -p, -r, -s, -y
Can be passed to the **delta** command.

-b, -c, -i, -l, -s, -x
Can be passed to the **get** command.

deledit Equivalent to the **delget** pseudo-command, except that the **get** portion of the sentence includes the **-e** flag. The **deledit** pseudo-command is useful for creating a checkpoint in your current editing session.

Flags:

-m, -p, -r, -s, -y
Can be passed to the **delta** command.

-b, -c, -i, -l, -s, -x
Can be passed to the **get** command.

create Creates an SCCS file, copying the initial contents from a file of the same name. If the file is successfully created, the original file is renamed with a comma on the front. You do not have to move or remove the original file as with the **admin** command.

Flags:

Accepts the same flags as the **admin** command. The **-i** flag is implied.

fix Removes a named delta, but leaves a copy of the delta with changes intact. This pseudo-command is useful for fixing small compiler errors. This pseudo-command does not keep a record of changes made to the file.

Flags:

-r*SID* Indicates a required flag.

clean Removes all files from the current directory or from the designated directory that can be recreated from SCCS files. Does not remove files that are in the process of being edited.

Flags:

-b Ignores branches when determining which files are being edited. Branches being edited in the same directory can be lost.

unedit Equivalent to the **unget** command. Any changes made since the **get** command was used are lost.

info Lists all files being edited.

Flags:

-b Ignores branches when determining which files are being edited.

-u [*Argument***]**
Lists only the files being edited by you or the user named by the *Argument* parameter.

check Prints all files being edited. Returns a nonzero exit status if a file is being edited. The check program can be used in a makefile to ensure that files are complete before a version is installed. Check the return code before performing the install.

Flags:

-b Ignores branches when determining which files are being edited.

-u [*Argument***]**
Lists only the files being edited by you or the user named by the *Argument* parameter.

tell Lists all files being edited, with a new line after each entry, on standard output.

Flags:

-b Ignores branches when determining which files are being edited.

-u [*Argument***]**
Lists only the files being edited by you or the user named by the *Argument* parameter.

diffs Shows the difference between the current version of the program you are editing and the previous deltas.

Flags:

-r, -c, -i, -x, -t
Can be passed to the **get** command.

-l, -s, -e, -f, -h, -b
Can be passed to the **diff** (not **sccsdiff**) command.

-C Can be passed to the **diff** (not **sccsdiff**) command as a **-c** flag.

print (*filename*(*s*)) Prints **verbose** information about the named files.

If the **PROJECTDIR** environment variable is set, its value determines the working directory. If this value begins with a / (slash), it is used directly. Otherwise, the value is interpreted as a user name whose home directory is examined for a subdirectory named **src** or **source**. If found, that subdirectory is used as the working directory.

Flags

-d*Path* Specifies a working directory for the SCCS files. The default is the current directory. The **-d** flag is prefixed to the entire path name of a file. When the **PROJECTDIR** environment variable is set and the **-d** flag is used, the command line overrides the environment value in determining the working directory.

-p Specifies a path name for the SCCS files. The default is the SCCS directory. The **-p** flag is inserted before the final component of the path name.

All flags specified after the command are passed to that command during execution. For a description of command flags, see the appropriate command description.

Example:

```
sccs -d/x -py get a/b
```

converts to:

```
get /x/a/y/s.b
```

This option is used to create aliases. For example:

```
alias syssccs sccs -d/usr/src
```

causes the **syssccs** command to become an alias command that can be used as follows:

```
syssccs get cmd/who.c
```

When used in this context, the above command will check the **/usr/src/cmd/SCCS** directory for the **s.who.c** file.

-r Runs the **sccs** command as the real user instead of as the effective user to which the **sccs** command is set (using the **set user id** command).

Certain commands, such as the **admin** command, cannot be run as **set user id**, which would allow anyone to change the authorizations. Such commands are always run as the real user.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

1. To get a file for editing, edit it, and then produce a new delta, enter:

   ```
   sccs get -e file.c
   ex file.c
   sccs delta file.c
   ```

2. To get a file from another directory, enter:

   ```
   sccs -p/usr/src/sccs/ get cc.c
   ```

   OR

   ```
   sccs get /usr/src/sccs/s.cc.c
   ```

3. To get a list of files being edited that are not on branches, enter:

   ```
   sccs info -b
   ```

Files

/usr/bin/sccs Contains the **sccs** command, which is the administration program for the SCCS commands.

Related Information

The **admin** command, **cdc** command, **comb** command, **delta** command, **diff** command, **get** command, **prs** command, **rmdel** command, **sact** command, **sccsdiff** command, **sccshelp** command, **unget** command, **val** command, **vc** command, **what** command.

The **sccsfile** file format.

List of SCCS Commands, Source Code Control System (SCCS) Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

sccsdiff Command

Purpose

Compares two versions of a SCCS file.

Syntax

sccsdiff -r*SID1* **-r***SID2* [**-p**] [**-s***Number*] *File* ...

Description

The **sccsdiff** command reads two versions of an Source Code Control System (SCCS) file, compares them, and then writes to standard output the differences between the two versions. Any number of SCCS files can be specified, but the same arguments apply to all files.

Flags

-p	Pipes the output through the **pr** command.
-r*SID1*	Specifies *SID1* as one delta of the SCCS file for the **sccsdiff** command to compare.
-r*SID2*	Specifies *SID2* as the other delta of the SCCS file for the **sccsdiff** command to compare.
-s*Number*	Specifies the file-segment size for the **bdiff** command to pass to the **diff** command. This is useful when the **diff** command fails due to a high system load.

Examples

To display the difference between versions 1.1 and 1.2 of SCCS file `s.test.c`, enter:

```
sccsdiff -r1.1 -r1.2 s.test.c
```

Files

/usr/bin/sccsdiff	Contains the SCCS **sccsdiff** command. The **sccsdiff** command supports multibyte character set (MBCS) data for the file names.

Related Information

The **bdiff** command, **diff** command, **get** command, **prs** command, **sccshelp** command.

The **sccsfile** file format.

List of SCCS Commands, Source Code Control System (SCCS) Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

sccshelp Command

Purpose

Provides information about a SCCS message or command.

Syntax

sccshelp [*ErrorCode*] [*Command*]

Description

The **sccshelp** command displays information about the use of a specified Source Code Control System (SCCS) command or about messages generated while using the commands. Each message has an associated code, which can be supplied as part of the argument to the **sccshelp** command. Zero or more arguments may be supplied. If you do not supply an argument, the **sccshelp** command prompts for one. You may include any of the SCCS commands as arguments to the **sccshelp** command.

The *ErrorCode* parameter specifies the code, consisting of numbers and letters, that appears at the end of a message. For example, in the following message, (cm7) is the code:

```
There are no SCCS identification keywords in the file. (cm7)
```

Examples

To get **sccshelp** on the **rmdel** command and two error codes, enter:

```
$ sccshelp rmdel gee ad3
```

The **sccshelp** command replies:

```
rmdel:
rmdel -r<SID> <file> ...
ERROR:
1255-141 gee is not a valid parameter. Specify a valid command or error code.
ad3:
The header flag you specified is not recognized.
The header flag you supplied with the -d or the -f flag is not correct.
Choose a valid header flag.
```

File

/usr/bin/sccshelp	Contains the SCCS **sccshelp** command.

Related Information

The **admin** command, **cdc** command, **comb** command, **delta** command, **get** command, **prs** command, **rmdel** command, **sccsdiff** command, **what** command.

The **sccsfile** file format.

List of SCCS Commands, Source Code Control System (SCCS) Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

schedo Command

Purpose

Manages CPU scheduler tunable parameters.

Syntax

schedo [**-p** | **-r**] { **-o** *Tunable*[**=***Newvalue*]}

schedo [**-p** | **-r**] { **-d** *Tunable* }

schedo [**-p** | **-r**] **-D**

schedo [**-p** | **-r**] **-a**

schedo -h [*Tunable*]

schedo -L [*Tunable*]

schedo -x [*Tunable*]

Note: Multiple flags **-o**, **-d**, **-x** and **-L** flags are allowed

Description

Note: The **schedo** command can only be executed by root.

Use the **schedo** command to configure scheduler tuning parameters. This command sets or displays current or next boot values for all scheduler tuning parameters. This command can also make permanent changes or defer changes until the next reboot. Whether the command sets or displays a parameter is determined by the accompanying flag. The **-o** flag performs both actions. It can either display the value of a parameter or set a new value for a parameter.

Understanding the Effect of Changing Tunable Parameters

Misuse of this command can cause performance degradation or operating-system failure. Be sure that you have studied the appropriate tuning sections in the *Performance management* before using **schedo** to change system parameters.

Before modifying any tunable parameter, you should first carefully read about all its characteristics in the Tunable Parameters section below, and follow any Refer To pointer, in order to fully understand its purpose.

You must then make sure that the Diagnosis and Tuning sections for this parameter truly apply to your situation and that changing the value of this parameter could help improve the performance of your system.

If the Diagnosis and Tuning sections both contain only ″N/A″, you should probably never change this parameter unless specifically directed by AIX development.

Priority-Calculation Parameters

The priority of most user processes varies with the amount of CPU time the process has used recently. The CPU scheduler's priority calculations are based on two parameters that are set with **schedo**, *sched_R* and *sched_D*. The *sched_R* and *sched_D* values are in thirty-seconds (1/32); that is, the formula used by the scheduler to calculate the amount to be added to a process's priority value as a penalty for recent CPU use is:

```
CPU penalty = (recently used CPU value of the process) * (r/32)
```

and the once-per-second recalculation of the recently used CPU value of each process is:

```
new recently used CPU value = (old recently used CPU value of the process) * (d/32)
```

Both r (*sched_R* parameter) and d (*sched_D* parameter) have default values of 16. This maintains the CPU scheduling behavior of previous versions of the operating system. Before experimenting with these values, you should be familiar with ″Tuning the CPU scheduler″ in the Performance Management Guide.

Memory-Load-Control Parameters

The operating system scheduler performs memory load control by suspending processes when memory is over committed. The system does not swap out processes; instead pages are *stolen* as they are needed to fulfill the current memory requirements. Typically, pages are stolen from suspended processes. Memory is considered over committed when the following condition is met:

*p * h s* where:

p is the number of pages written to paging space in the last second
h is an integer specified by the v_repage_hi parameter
s is the number of page steals that have occurred in the last second

A process is suspended when memory is over committed and the following condition is met:

*r * p f* where:

r r is the number of repages that the process has accumulated in the last second
p is an integer specified by the v_repage_proc parameter
f is the number of page faults that the process has experienced in the last second

In addition, fixed-priority processes and kernel processes are exempt from being suspended.

The term repages refers to the number of pages belonging to the process, which were reclaimed and are soon after referenced again by the process.

The user also can specify a minimum multiprogramming level with the v_min_process parameter. Doing so ensures that a minimum number of processes remain active throughout the process-suspension period. Active processes are those that are runnable and waiting for page I/O. Processes that are waiting for events and processes that are suspended are not considered active, nor is the wait process considered active.

Suspended processes can be added back into the mix when the system has stayed below the over committed threshold for n seconds, where n is specified by the v_sec_wait parameter. Processes are added back into the system based, first, on their priority and, second, on the length of their suspension period.

Before experimenting with these values, you should be thoroughly familiar with ″VMM memory load control tuning with the schedo command″ in the Performance Management Guide.

Time-Slice-Increment Parameter

The **schedo** command can also be used to change the amount of time the operating system allows a given process to run before the dispatcher is called to choose another process to run (the time slice). The default value for this interval is a single clock tick (10 milliseconds). The timeslice tuning parameter allows the user to specify the number of clock ticks by which the time slice length is to be increased.

In AIX Version 4, this parameter only applies to threads with the SCHED_RR scheduling policy. See Scheduling Policy for Threads.

fork() Retry Interval Parameter

If a **fork**() subroutine call fails because there is not enough paging space available to create a new process, the system retries the call after waiting for a specified period of time. That interval is set with the pacefork tuning parameter.

Special Terminology for Symmetric Multithreading

Multiple run queues are supported. Under this scheme each CPU has it's own run queue. Power 5 processors support symmetric multithreading, where each physical processor has two execution engines, called *hardware threads*. Each hardware thread is essentially equivalent to a single CPU. Symmetric multithreading is enabled by default, but it can be disabled (or re-enabled) dynamically. When symmetric multithreading is enabled, each hardware thread services a separate run queue. For example, on a 4-way system when symmetric multithreading is disabled or not present, there are 4 run queues in addition to the global run queue. When symmetric multithreading is enabled, there are 8 run queues in addition to the global run queue.

The hardware threads belonging to the same physical processor are referred to as *sibling threads*. A *primary sibling thread* is the first hardware thread of the physical processor. A *secondary sibling thread* is the second hardware thread of the physical processor.

Virtual Processor Management

More virtual processors can be defined than are needed to handle the work in a partition. The overhead of dispatching virtual processors can be reduced by using fewer virtual processors without a decrease in overall processor usage or a lack of virtual processors. Virtual processors are not dynamically removed from the partition, but instead are not used and are used again only when additional work is available. Each virtual processor uses a maximum of one physical processor. The number of virtual processes needed is determined by rounding up the sum of the physical processor utilization and the **vpm_xvcpus** tunable:

```
number = ceiling( p_util + vpm_xvcpus)
```

Where *number* is the number of virtual processors that are needed, *p_util* is the physical processor utilization, and **vpm_xvcpus** is a tunable that specifies the number of additional virtual processors to enable. If *number* is less than the number of currently enabled virtual processors, a virtual processor will be disabled. If *number* is greater than the number of currently enabled virtual processors, a disabled virtual processor will be enabled. Threads that are attached to a disabled virtual processor are still allowed to run on the disabled virtual processor.

Node Load

The *node load*, or simply *load*, is the average run queue depth across all run queues, including the global run queue multiplied by 256, and is strongly smoothed over time. For example, a load of 256 means that if we have 16 CPUs (including symmetric multithreading CPUs), then we have had approximately 16 runnable jobs in the system for the last few milliseconds.

Flags

-a	Displays the current, reboot (when used in conjunction with **-r**) or permanent (when used in conjunction with **-p**) value for all tunable parameters, one per line in pairs *Tunable* = *Value*. For the permanent option, a value is only displayed for a parameter if its reboot and current values are equal. Otherwise `NONE` displays as the value.
-d *Tunable*	Resets *Tunable* to its default value. If a tunable needs to be changed (that is, it is currently not set to its default value, and **-r** is not used in combination, it won't be changed but a warning is displayed.
-D	Resets all tunables to their default value. If tunables needing to be changed are of type Bosboot or Reboot, or are of type Incremental and have been changed from their default value, and **-r** is not used in combination, they will not be changed but a warning displays.
-h [*Tunable*]	Displays help about the *Tunable* parameter if one is specified. Otherwise, displays the **schedo** command usage statement.

-L [*Tunable*] Lists the characteristics of one or all tunables, one per line, using the following format:

```
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT           TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
v_repage_hi               0      0      0      0      2047M                   D
------------------------------------------------------------------------------
v_repage_proc             4      4      4      0      2047M                   D
------------------------------------------------------------------------------
v_sec_wait                1      1      1      0      2047M  seconds          D
------------------------------------------------------------------------------
...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot),
           B (for Bosboot), M (for Mount), I (for Incremental), C (for Connect), and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

-o *Tunable* [**=***Newvalue*] Displays the value or sets *Tunable* to *Newvalue*. If a tunable needs to be changed (the specified value is different than current value), and is of type Bosboot or Reboot, or if it is of type Incremental and its current value is bigger than the specified value, and **-r** is not used in combination, it will not be changed but a warning displays.

When **-r** is used in combination without a new value, the nextboot value for tunable is displayed. When **-p** is used in combination without a new value, a value displays only if the current and next boot values for tunable are the same. Otherwise NONE displays as the value.

-p Makes changes apply to both current and reboot values, when used in combination with **-o**, **-d** or **-D**, that is, turns on the updating of the **/etc/tunables/nextboot** file in addition to the updating of the current value. These combinations cannot be used on Reboot and Bosboot type parameters because their current value can't be changed.

When used with **-a** or **-o** without specifying a new value, values are displayed only if the current and next boot values for a parameter are the same. Otherwise NONE displays as the value.

-r Makes changes apply to reboot values when used in combination with **-o**, **-d** or **-D**, that is, turns on the updating of the **/etc/tunables/nextboot** file. If any parameter of type Bosboot is changed, the user will be prompted to run bosboot.

When used with **-a** or **-o** without specifying a new value, next boot values for tunables display instead of current values.

-x [*Tunable*] Lists characteristics of one or all tunables, one per line, using the following (spreadsheet) format:

```
tunable,current,default,reboot,min,max,unit,type,{dtunable }

where:
    current = current value
    default = default value
    reboot = reboot value
    min = minimal value
    max = maximum value
    unit = tunable unit of measure
    type = parameter type: D (for Dynamic), S (for Static), R (for Reboot),
               B (for Bosboot),M (for Mount), I (for Incremental),
               C (for Connect), and d (for Deprecated)
    dtunable = space separated list of dependent tunable parameters
```

Any change (with **-o**, **-d** or **-D**) to a parameter of type Mount results in a message displaying to warn the user that the change is only effective for future mountings.

Any change (with **-o**, **-d** or **-D** flags) to a parameter of type Connect will result in **inetd** being restarted, and in a message being displayed to warn the user that the change is only effective for future socket connections.

Any attempt to change (with **-o**, **-d** or **-D**) a parameter of type Bosboot or Reboot without **-r**, results in an error message.

Any attempt to change (with**-o**, **-d** or **-D** but without **-r**) the current value of a parameter of type Incremental with a new value smaller than the current value, results in an error message.

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, **raso**, and **schedo**) have been classified into these categories:

Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can only be changed during reboot
Bosboot	If the parameter can only be changed by running bosboot and rebooting the machine
Mount	If changes to the parameter are only effective for future file systems or directory mounts
Incremental	If the parameter can only be incremented, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

For parameters of type Bosboot, whenever a change is performed, the tuning commands automatically prompt the user to ask if they want to execute the **bosboot** command. For parameters of type Connect, the tuning commands automatically restart the **inetd** daemon.

Note that the current set of parameters managed by the **schedo** command only includes Dynamic, and Reboot types.

Compatibility Mode

When running in pre 5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys**0, see AIX 5.2 compatibility mode in the *Performance management*), reboot values for parameters, except those of type Bosboot, are not really meaningful because in this mode they are not applied at boot time.

In pre 5.2 compatibility mode, setting reboot values to tuning parameters continues to be achieved by imbedding calls to tuning commands in scripts called during the boot sequence. Parameters of type **Reboot** can therefore be set without the **-r** flag, so that existing scripts continue to work.

This mode is automatically turned ON when a machine is MIGRATED to AIX 5.2. For complete installations, it is turned OFF and the reboot values for parameters are set by applying the content of the **/etc/tunables/nextboot** file during the reboot sequence. Only in that mode are the **-r** and **-p** flags fully functional. See Kernel Tuning in the *AIX 5L Version 5.3 Performance Tools Guide and Reference* for more information.

Tunable Parameters

affinity_lim

Purpose:

Sets the number of intervening dispatches after which the SCHED_FIFO2 policy no longer favors a thread.

Values:

Default: 7

Range: 0 to 100

Type: Dynamic

Diagnosis:

N/A

Tuning:

Once a thread is running with SCHED_FIFO2 policy, tuning of this variable may or may not have an effect on the performance of the thread and workload. Ideal values should be determined by trial and error.

Refer To:

Scheduling policy for threads

allowMCMmigrate

Purpose:

Enables (1) or disables (0) the weakening of barriers for migration threads between MCMs under light load conditions.

Values:

Default: 0 (disabled)

Range: 0 to 1

Type: Boolean

Diagnosis:

N/A

Tuning:

N/A

big_tick_size

Purpose:

Sets physical tick interval & synchronizes ticks across cpus.

Values:

Default: 1

Range: 1 to 100

Type: Dynamic

Diagnosis:

N/A

Tuning:

This value multiplied by 10 ms is the tick interval, and should evenly divide into 100. Use of this parameter will make system statistics less accurate.

fixed_pri_global

Purpose:
Keep fixed priority threads on global run queue.

Values:
Default: 0
Range: 0 to 1
Type: Dynamic

Diagnosis:
N/A

Tuning:
If 1, then fixed priority threads are placed on the global run queue.

Refer To:
Scheduler run queue

force_grq

Purpose:
Keep non-MPI threads on the global run queue.

Values:
Default: 0
Range: 0 or 1
Type: Dynamic

Diagnosis:
N/A

Tuning:
If set to 1, only MPI and bound threads will use local run queues, which may hurt performance.

hotlocks_enable

Purpose:
Enables (1) or disables (0) the hardware priority boosting of hot locks.

Values:
Default: 0 (disabled)
Range: 0...1
Type: Dynamic

Diagnosis:
N/A

Tuning:
N/A

idle_migration_barrier

Purpose:
Used to determine when threads can be migrated to other processors.

Values:
Default: 4
Range: 0 to 100
Type: Dynamic

Diagnosis:
N/A

Tuning:
This value is divided by 16 and then multiplied by the load average. The resulting value is used to determine if jobs should be migrated to other nodes (essentially does load balancing).

krlock_confer2self

Purpose:

Enables (1) or disables (0) conferring to self after trying to acquire krlock krlock_spinb4confer times. This parameter only applies to the 64-bit kernel.

Values:

Default: 0 (disabled)

Range: 0...1

Type: Dynamic (Reboot when 32-bit kernel is running.)

Diagnosis:

N/A

Tuning:

N/A

krlock_conferb4alloc

Purpose:

Enables (1) or disables (0) conferring after spinning **slock_spinb4confer** before trying to acquire or allocating krlock. This parameter only applies to the 64-bit kernel.

Values:

Default: 0 (disabled)

Range: 0...1

Type: Dynamic (Reboot when 32-bit kernel is running.)

Diagnosis:

N/A

Tuning:

N/A

krlock_enable

Purpose:

Enables (1) or disables (0) krlocks. This parameter only applies to the 64-bit kernel.

Values:

Default: 1 (enabled)

Range: 0...1

Type: Dynamic (Reboot when 32-bit kernel is running.)

Diagnosis:

N/A

Tuning:

N/A

krlock_spinb4alloc

Purpose:

Number of additional acquisition attempts after spinning **slock_spinb4confer**, and conferring (if **krlock_conferb4alloc** is on), before allocating krlock. This parameter only applies to the 64-bit kernel.

Values:

Default: 1

Range: 1...MAXINT

Type: Dynamic (Reboot when 32-bit kernel is running.)

Diagnosis:

N/A

Tuning:

N/A

krlock_spinb4confer

Purpose:

Number of krlock acquisition attempts before conferring to the krlock holder (or self). This parameter only applies to the 64-bit kernel.

Values:

Default: 1024

Range: 0...MAXINT

Type: Dynamic (Reboot when 32-bit kernel is running.)

Diagnosis:

N/A

Tuning:

N/A

maxspin

Purpose:

Sets the number of times to spin on a kernel lock before going to sleep.

Values:

Default: 1 on uniprocessor systems, -1 on MP systems, which means to spin up to 2^{32} times

Range: -1 to 2^{32}

Type: Dynamic

Diagnosis:

N/A

Tuning:

Increasing the value or setting it to -1 on MP systems may reduce idle time; however, it may also waste CPU time in some situations. Increasing it on uniprocessor systems is not recommended.

Refer To:

Use of the schedo command to modify the MAXSPIN parameter

n_idle_loop_vlopri

Purpose:

Number of times to run the low hardware priority loop each time in idle loop if no new work is found.

Values:

Default: 100

Range: 0...1000000

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

pacefork

Purpose:

The number of clock ticks to wait before retrying a failed fork call that has failed for lack of paging space.

Values:

Default: 10

Range: a positive number of clock ticks bigger than 10

Type: Dynamic

Diagnosis:

System is running out of paging space and process cannot be forked.

Tuning:

The system will retry a failed fork five times. For example, if a **fork**() subroutine call fails because there is not enough paging space available to create a new process, the system retries the call after waiting the specified number of clock ticks.

Refer To:

The fork() retry interval parameter

sched_D

Purpose:

Sets the short term CPU usage delay rate.

Values:

Default: 16

Range: 0 to 32

Type: Dynamic

Diagnosis:

N/A

Tuning:

The default is to decay short-term CPU usage by 1/2 (16/32) every second. Decreasing this value enables foreground processes to avoid competition with background processes for a longer time.

Refer To:

Thread-Priority-Value calculation

sched_R

Purpose:

Sets the weighting factor for short-term CPU usage in priority calculations.

Values:

Default: 16

Range: 0 to 32

Type: Dynamic

Diagnosis:

Run: **ps al**. If you find that the PRI column has priority values for foreground processes (those with NI values of 20) that are higher than the PRI values of some background processes (NI values > 20), you can reduce the r value.

Tuning:

The default is to include 1/2 (16/32) of the short term CPU usage in the priority calculation. Decreasing this value makes it easier for foreground processes to compete.

Refer To:

Thread-Priority-Value calculation

search_globalrq_mload

Purpose:

Minimum load above which secondary sibling threads will look for work in the global run queue in the dispatcher.

Values:

Default: 256

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

search_smtrunq_mload

Purpose:

Minimum load above which the dispatcher will also search the run queues belonging to its sibling hardware threads. This is meant for load balancing on a physical processor and is not the same as idle load balancing as this check is made in the dispatcher when choosing the next job to be dispatched. This works in conjunction with the smtrunq_load_diff tunable.

Values:

Default: 256

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

setnewrq_sidle_mload

Purpose:

Minimum system load above which idle secondary sibling threads will be considered for new work even when primary is not idle.

Values:

Default: 384

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

shed_primrunq_mload

Purpose:

The maximum load below which the secondary sibling threads will try to shed work onto the primary sibling thread's run queue.

Values:

Default: 64

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

sidle_s1runq_mload

Purpose:

The minimum load above which idle load balancing for secondary sibling threads will search for work in the primary sibling thread's run queue.

Values:

Default: 64

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

sidle_S2runq_mload

Purpose:

Minimum load above which secondary sibling threads will look for work among other run queues owned by CPUs within their S2 affinity domain during idle load balancing.

Values:

Default: 134

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

It is recommended that this tunable never be set to a value that is less than the value of sidle_S1runq_mload.

sidle_S3runq_mload

Purpose:

Minimum load above which secondary sibling threads will look for work among other run queues owned by CPUs within their S3 affinity domain during idle load balancing.

Values:

Default: 134

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

It is recommended that this tunable never be set to a value that is less than the value of sidle_S2runq_mload.

sidle_S4runq_mload

Purpose:

Minimum load above which secondary sibling threads will look for work on any local run queues.

Values:

Default: 4294967040

Range: 0...4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

It is recommended that this tunable never be set to a value that is less than the value of sidle_S3runq_mload.

slock_spinb4confer

Purpose:

Number of attempts for a simple lock before conferring.

Values:

Default: 1024

Range: 0...MAXINT

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

smt_snooze_delay

Purpose:

Amount of time in microseconds in idle loop without useful work before snoozing (calling **h_cede**). A value of `-1` indicates to disable snoozing, a value of `0` indicates to snooze immediately.

Values:

Default: 0

Range: -1..100000000 (100 secs)

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

smtrunq_load_diff

Purpose:

Minimum load difference between dibling run queue loads for a task to be stolen from the sibling's run queue. This is enabled only when the load is greater than the value for the search_smtrunq_mload tunable.

Values:

Default: 2

Range: 1..4294967040

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

tb_balance_s0

Purpose:

Controls SMT-cores busy balancing. This balancing attempts to keep compute-bound threads spread across physical resources.

Values:

Default: 0 (balancing disabled)

Range: 0, 1, or 2. A value of 1 indicates balancing enabled within MCMs (S2 groups).

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

tb_balance_s1

Purpose:

Controls chipset busy balancing. This balancing attempts to keep compute-bound threads spread across physical resources.

Values:

Default: 1 (balancing enabled system-wide)

Range: 0, 1, or 2. A value of 0 indicates balancing disabled. A value of 1 indicates balancing enabled within MCMs (S2 groups).

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

tb_threshold

Purpose:

Number of ticks to consider a thread busy for optimization purposes for thread_busy load balancing. This balancing attempts to keep compute-bound threads spread across physical resources.

Values:

Default: 100 (1 second)

Range: 10 to 1000 (0.1 to 10 seconds)

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

timeslice

Purpose:

The number of clock ticks a thread can run before it is put back on the run queue.

Values:

Default: 1

Range: a positive integer value

Type: Dynamic

Diagnosis:

N/A

Tuning:

Increasing this value can reduce overhead of dispatching threads. The value refers to the total number of clock ticks in a timeslice and only affects fixed-priority processes.

Refer To:

Modification of the scheduler time slice with the schedo command

unboost_inflih

Purpose:

Enables (1) or disables (0) the unboost of the hot lock priority in the **flih**. When disabled, the unboost occurs in the dispatcher.

Values:

Default: 1 (enabled)

Range: 0...1

Type: Dynamic

Diagnosis:

N/A

Tuning:

N/A

%usDelta

Purpose:

Used to adjust system clock with each clock tick in the correction range -1 to +1 seconds.

Values:

Default: 100

Range: 0 to 100

Type: Dynamic

Diagnosis:

N/A

Tuning:

This is used to adjust clock drifts.

v_exempt_secs

Purpose:

Sets the number of seconds that a recently resumed process that was previously suspended is exempt from suspension.

Values:

Default: 2

Range: 0 or a positive number

Type: Dynamic

Diagnosis:

N/A

Tuning:

This parameter is only examined if thrashing is occurring.

Refer To:

VMM memory load control facility and VMM memory load control tuning with the schedo command.

v_min_process

Purpose:

Sets the minimum number of processes that are exempt from suspension.

Values:

Default: 2

Range: 0 or a positive number

Type: Dynamic

Diagnosis:

N/A

Tuning:

This number is in addition to kernel processes, processes with fixed priority less than 60, processes with pinned memory, or processes awaiting events. This parameter is only examined if there are threads on the suspended queue.

Refer To:

VMM memory load control facility and VMM memory load control tuning with the schedo command.

v_repage_hi

Purpose:

Sets the system wide criteria used to determine when process suspension begins and ends (system is thrashing).

Values:

Default: 6 unless system RAM is 128 MB or more (in this case it is 0)

Range: 0 or a positive number

Type: Dynamic

Diagnosis:

If v_repage_hi * page_outs/sec is > page_steals, then processes may get suspended.

Tuning:

If system is paging and causing scheduler to think it is thrashing but thrashing is not actually occurring, then it may be useful to desensitize the algorithm by decreasing the **-h** value or setting it to 0.

Refer To:

VMM memory load control facility and VMM memory load control tuning with the schedo command.

v_repage_proc

Purpose:
Sets the per-process criterion used to determine which processes to suspend.

Values:
Default: 4
Range: 0 or a positive number
Type: Dynamic

Diagnosis:
N/A

Tuning:
This requires a higher level of repaging by a given process before it is a candidate for suspension by memory load control. This parameter is examined only if thrashing is occurring.

Refer To:
VMM memory load control facility and VMM memory load control tuning with the schedo command.

v_sec_wait

Purpose:
Sets the number of seconds to wait after thrashing ends before making suspended processes runnable.

Values:
Default: 1
Range: 0 or a positive number
Type: Dynamic

Diagnosis:
N/A

Tuning:
This parameter is examined only if thrashing is occurring.

Refer To:
VMM memory load control facility and VMM memory load control tuning with the schedo command.

vpm_xvcpus

Purpose:
Specifies the number of virtual processors to enable in addition to the number of virtual processors that are needed to consume the physical CPU utilization.

Values:
Default: 0 (on)
Range: -1 to **INT_MAX**
Type: Dynamic

Diagnosis:
N/A

Tuning:
A value of **-1** disables this functionality.

Examples

1. To list the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the schedo command, type:

```
schedo -L
```

2. To list (spreadsheet format) the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the **schedo** command, type:
   ```
   schedo -x
   ```
3. To reset v_sec_wait to default, type:
   ```
   schedo -d v_sec_wait
   ```
4. To display help on sched_R, type:
   ```
   schedo -h sched_R
   ```
5. To set v_min_process to 4 after the next reboot, type:
   ```
   schedo -r -o v_min_process=4
   ```
6. To permanently reset all schedo tunable parameters to default, type:
   ```
   schedo -p -D
   ```
7. To list the reboot value for all schedo parameters, type:
   ```
   schedo -r -a
   ```

Related Information

The **vmo** command, **ioo** command, **no** command, **nfso** command, **raso** command, **tunchange** command, **tunsave** command, **tunrestore** command, **tuncheck** command, and **tundefault** command.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*

AIX 5.2 compatibility mode in *Performance management*.

---

scls Command

Purpose

Produces a list of module and driver names.

Syntax

scls [**-c** | **-l**] [*Module* ...]

Description

The **scls** command provides a method for the user to query the current Portable Streams Environment (PSE) configuration. The **scls** command produces a list of module and driver names. Flags can be used to produce enhanced lists. Any further parameters on the command line are module or driver names, and the output produced is for only those names.

> **Note:** The **scls** command requires the **sc** STREAMS module and the **nuls** driver. If either one is not available, the **scls** command will not be successful.

Flags

-c Produces a listing showing the number of times an interface routine was called.
-l Produces a long listing that shows the extension type, major number, and information pertaining to the **module_info** structure.

The **-c** and **-l** flags are mutually exclusive.

Parameters

module Specifies the name of the modules or drivers for which to output information.

Files

sc Dynamically loadable STREAMS configuration module
nuls Dynamically loadable STREAMS null device.

Related Information

The **strload** command.

List of Streams Commands in *AIX 5L Version 5.3 Communications Programming Concepts*.

Configuring Drivers and Modules in the Portable Streams Environment (PSE), STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

---

script Command

Purpose

Makes a typescript of a terminal session.

Syntax

script [**-a**] [**-q**] [*File*]

Description

The **script** command makes a typescript of everything displayed on your terminal. The typescript is written to the file specified by the *File* parameter. The typescript can later be sent to the line printer. If no file name is given, the typescript is saved in the current directory with the file name **typescript**.

The script ends when the forked shell exits.

This command is useful for producing hardcopy records when hardcopy terminals are in short supply. For example, use the **script** command when you are working on a CRT display and need a hardcopy record of the dialog.

Because the **script** command sets the **SetUserID** mode bit, due to security reasons the value of LIBPATH variable is unset when the command is invoked. However, LIBPATH is automatically reset in the forked shell if it is defined in the environment file. This behavior is also true for the NLSPATH environment variable. For related information, see the **exec** subroutine.

Flags

-a Appends the typescript to the specified file or to the **typescript** file.
-q Suppresses diagnostic messages.

Files

/usr/bin/script Contains the **script** command.

Related Information

The **tee** command.

Input and output redirection in *Operating system and device management* describes how the operating system processes input and output and how to use the redirect and pipe symbols.

---

sctpctrl Command

Purpose

Controls and configures SCTP.

Syntax

sctpctrl {*load*|*unload*|*set*}

sctpctrl *stats* [*reset*] [*interval*]

sctpctrl *set* {*name*=*value*|*default* [*name*]}

sctpctrl *get* [*name*]

Description

The **sctpctrl** command is used to control and configure the SCTP kernel extension. This command can be used to load and unload the SCTP kernel extension. This can also be used to dump SCTP data and set or retrieve various SCTP tunable. Further, **sctpctrl** command can be used to read and reset the SCTP specific network statistics.

Parameters

load	Loads the SCTP kernel extension if not loaded.
unload	Unloads the SCTP kernel extension if loaded
stats [*reset*] [*interval*]	Displays SCTP statistics. The optional **reset** command will clear (zero) the statistics. If the *interval* parameter (in seconds) is added, the program does not exit, but outputs the statistics every [*interval*] seconds.
set {*name*=*value*|*default* [*name*]}	Sets the SCTP tunable to a value. If *default* is specified then it will set all the tunable to their default values. If optional [*name*] is specified followed by *default* then it will set tunable described by *name* to its default value.
get [*name*]	Gets the value of the tunable described by their optional *name* parameter. If *name* parameter is not specified then it gets the values of all the tunable.

Examples

1. To load the SCTP kernel extension, type the following:
   ```
   sctpctrl load
   ```
2. To unload SCTP kernel extension, type the following:
   ```
   sctpctrl unload
   ```
3. To reset the SCTP statistics, type the following:
   ```
   sctpctrl stats reset
   ```

   This command will zero-out all the SCTP statistics.

4. To get the values of the SCTP tunable, type the following:

```
sctpctrl get
```

   This will list all the SCTP tunable and their values. Here is a sample output.

```
sctp_assoc_maxerr = 10
sctp_cookie_life = 60
sctp_delack_timer = 4
sctp_dontdelayack = 1
sctp_ecn = 1
sctp_ephemeral_high = 65535
sctp_ephemeral_low = 32768
sctp_instreams = 2048
sctp_maxburst = 8
sctp_outstreams = 10
sctp_path_maxerr = 5
sctp_pmtu_discover = 1
sctp_rttmax = 60
sctp_rttmin = 1
sctp_recvspace = 65536
sctp_sendspace = 65536
sctp_send_fewsacks = 0
```

5. To set **sctp_path_maxerr** to a value of 6, type the following:

```
sctpctrl set sctp_path_maxerr=6
```

Location

/usr/sbin/sctpctrl

Files

/usr/sbin/sctpctrl	Contains the **sctpctrl** command.
/usr/lib/drivers/sctp	Contains the SCTP kernel extension.

Related Information

The **sctp_peeloff** subroutine, **sctp_opt_info** subroutine.

Stream control transmission protocol in *Networks and communication management*.

---

sdiff Command

Purpose

Compares two files and displays the differences in a side-by-side format.

Syntax

sdiff [**-l** | **-s**] [**-o** *OutFile*] [**-w** *Number*] *File1 File2*

Description

The **sdiff** command reads the files specified by the *File1* and *File2* parameters, uses the **diff** command to compare them, and writes the results to standard output in a side-by-side format. The **sdiff** command displays each line of the two files with a series of spaces between them if the lines are identical. It displays a < (less than sign) in the field of spaces if the line only exists in the file specified by the *File1* parameter, a > (greater than sign) if the line only exists in the file specified by the *File2* parameter, and a | (vertical bar) for lines that are different.

When you specify the **-o** flag, the **sdiff** command merges the files specified by the *File1* and *File2* parameters and produces a third file.

Note: The **sdiff** command invokes the **diff -b** command to compare two input files. The **-b** flag causes the **diff** command to ignore trailing spaces and tab characters and to consider other strings of spaces as equal.

Flags

-l Displays only the left side when lines are identical.

-o *OutFile* Creates a third file, specified by the *OutFile* variable, by a controlled line-by-line merging of the two files specified by the *File1* and the *File2* parameters. The following subcommands govern the creation of this file:

e Starts the **ed** command with an empty file.

e b or e l
Starts the **ed** command with both sides.

e l or e <
Starts the **ed** command with the left side.

e r or e >
Starts the **ed** command with the right side.

l Adds the left side to the output file.

r Adds the right side to the output file.

s Stops displaying identical lines.

v Begins displaying identical lines.

q Performs one of the following functions:

- Exits the **ed** command.
- Exits the **sdiff** command if no **ed** command is running.
- Exits both commands. This action occurs when there are no more lines to be merged into the output file.

Each time you exit from the **ed** command, the **sdiff** command writes the resulting edited file to the end of the file specified by the *OutFile* variable. If you do not save the changes before exiting (for example, you press the Ctrl-C key sequence), the **sdiff** command writes the initial input to the output file.

-s Does not display identical lines.

-w *Number* Sets the width of the output line. The default value of the *Number* variable is 130 characters. The maximum width of the *Number* variable is 2048. The minimum width of the *Number* variable is 20. The **sdiff** command uses 2048 if a value greater than 2048 is specified.

Examples

1. To print a comparison of two files, enter:

   ```
   sdiff chap1.bak chap1
   ```

   The **sdiff** command displays a side-by-side listing that compares each line of the `chap1.bak` and `chap1` files.
2. To display only the lines that differ, enter:

   ```
   sdiff  -s  -w 80 chap1.bak chap1
   ```

The **sdiff** command displays the differences at the work station. The `-w 80` flag and variable sets the page width to 80 columns. The `-s` flag indicates lines that are identical in both files will not be displayed.

3. To selectively combine parts of two files, enter:

   ```
   sdiff  -s  -w 80  -o chap1.combo chap1.bak chap1
   ```

   The **sdiff** command combines the `chap1.bak` and `chap1` files into a new file called `chap1.combo`. For each group of differing lines, the **sdiff** command prompts you which group to keep or whether you want to edit them using the **ed** command.

4. To combine and edit two files, `staff.jan` and `staff.apr`, and write the results to the `staff.year` file, perform the steps indicated.

   The `staff.jan` file contains the following lines:

   ```
   Members of the Accounting Department
   Andrea
   George
   Karen
   Sam
   Thomas
   ```

   The `staff.apr` file contains the following lines:

   ```
   Members of the Accounting Department
   Andrea
   Fred
   Mark
   Sam
   Wendy
   ```

   a. Enter the following command:

      ```
      sdiff -o staff.year staff.jan staff.apr
      ```

      The **sdiff** command will begin to compare the contents of the `staff.jan` and `staff.apr` files and write the results to the `staff.year` file. The **sdiff** command displays the following:

      ```
      Members of the Accounting Dept   Members of the Accounting Dept
      Andrea                            Andrea
      George                          | Fred
      %
      ```

      The % (percent sign) is the command prompt.

   b. Enter the **e b** subcommand to start editing the output file with the **ed** command.

      The **sdiff** command displays a sequence of digits, indicating the byte count of lines being merged. In this case, the byte count is `23`.

   c. Enter the **q** subcommand to exit the **ed** command and continue combining and editing the two files. The **sdiff** command displays the following:

      ```
      Sam                               Sam
      Thomas                          | Wendy
      ```

   d. Enter the **e b** subcommand again. The **ed** command must be run each time a set of lines from the original two files are to be merged into the output file. The byte count in this instance is `13`.

   e. Enter the **q** subcommand to save the changes. When all the lines of the two files have been merged into the output file, the **q** subcommand exits the **ed** and **sdiff** commands.

   The `staff.year` file now contains the following:

   ```
   Members of the Accounting Department
   Andrea
   George
   Karen
   Fred
   ```

```
Mark
Sam
Thomas
Wendy
```

Files

/usr/bin/sdiff	Contains the **sdiff** command.

Related Information

The **diff** command, **ed** command.

Files, Input and output redirection in *Operating system and device management*.

---

secldapclntd Daemon

Purpose

Provides and manages connection and handles transactions between the LDAP load module and the LDAP Security Information Server.

Syntax

/usr/sbin/secldapclntd [**-C** *CacheSize*] [**-p** *NumOfThread*] [**-t** *CacheTimeOut*] [**-T** *HeartBeatIntv*] [**-o** *ldapTimeOut*]

Description

The **secldapclntd** daemon accepts requests from the LDAP load module, forwards the request to the LDAP Security Information Server, and passes the result from the server back to the LDAP load module. This daemon reads the configuration information defined in the **/etc/security/ldap/ldap.cfg** file during its startup, authenticates to the LDAP Security Information Server using the specified server distinguished name and password, and establishes a connection between the local host and the server.

If multiple servers are specified in the **/etc/security/ldap/ldap.cfg** file, the **secldapclntd** daemon connects to all of the servers. At a specific time, however, it talks to only one of them. The priority of the server connection is determined by its location in the server list with the highest priority server listed first. The **secldapclntd** daemon can detect when the server it is currently communicating with is down, and automatically switches to another available server. It can also detect when a server becomes available again and re-establish connection to that server. If the reconnected server is of higher priority then the current server then communication is switched to it. This auto-detect feature is done by the **secldapclntd** daemon checking on each of the servers periodically. The time interval between subsequent checking is defaulted to 300 seconds, and can be changed at the daemon startup time from the command line with the **-T** option or by modifying the **heartbeatinterval** value in the **/etc/ security/ldap/ldap.cfg** file.

At startup, the **secldapclntd** daemon tries to establish a connection to the LDAP servers. If it cannot connect to any of the servers, it goes to sleep, and tries again in 30 seconds. It repeats this process twice, and if it still cannot establish any connection, the **secldapclntd** daemon process exits.

The **secldapclntd** daemon is a multi-threaded program. The default number of threads used by this daemon is 10. An administrator can fine-tune the system performance by adjusting the number of threads used by this daemon.

The **secldapclntd** daemon caches information retrieved from the LDAP Security Information Server for performance purpose. If the requested data can be found in the cache and the cache entry is not expired,

the data in the cache is handed back to the requester. Otherwise, the **secldapclntd** daemon makes a request to the LDAP Security Information Server for the information.

The valid number of cache entries for users is in the range of 100-10,000, and that for groups is in the range of 10-1,000. The default is 1000 entries for users, and 100 entries for groups.

The cache timeout or TTL (time to live) can be from 60 seconds to 1 hour (60*60=3600 seconds). By default, a cache entry expires in 300 seconds. If the cache timeout is set to 0, the caching feature is disabled.

Communication between the **secldapclntd** daemon and the LDAP server is performed using asynchronous methods. This allows the daemon to request information from the server and then perform other steps while waiting for the request to return. The length of time that the client will wait for a response from a server is configurable by the administrator and defaults to 60 seconds.

When connecting to LDAP servers, the **secldapclntd** daemon needs to do host lookups. The **nis_ldap** resolver may cause the lookup to be routed back to the daemon itself, resulting in a hang situation. To avoid this problem, the **secldapclntd** daemon ignores the system order of name resolution. Instead, it uses the order defined by the **nsorder** attribute in the **/etc/security/ldap/ldap.cfg** file.

Flags

Note: By default, the **secldapclntd** daemon reads the configuration information specified in the **/etc/security/ldap/ldap.cfg** file at startup. If the following options are given on the command line when starting the **secldapclntd** process, the options from the command line will override the values in the **/etc/security/ldap/ldap.cfg** file.

-C *CacheSize*	Sets the maximum cache entries used by the **secldapclntd** daemon to *CacheSize* number of entries. Valid range is 100-10,000 entries for user cache. The default is 1000. The group cache entries will be 10% of the user cache entries.
-o *ldapTimeOut*	Timeout period in seconds for LDAP client requests to the server. This value determines how long the client will wait for a response from the LDAP server. Valid range is 0 - 3600 (1 hour). Default is 60 seconds. Set this value to 0 to disable the timeout and force the client to wait indefinitely.
-p *NumOfThread*	Sets the number of threads used by the **secldapclntd** daemon to *NumOfThread* threads. Valid range is 1-1000. The default is 10.
-t *CacheTimeout*	Sets the cache to expire in *CacheTimeout* seconds. Valid range is 60- 3600 seconds. The default is 300 seconds.
-T *HeartBeatIntv*	Sets the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.

Examples

1. To start the **secldapclntd** daemon, type:
   ```
   /usr/sbin/secldapclntd
   ```
2. To start the **secldapclntd** using 20 threads and cache timeout value of 600 seconds, type:
   ```
   /usr/sbin/secldapclntd -p 20 -t 600
   ```

Use of the **start-secldapclntd** command is recommended for starting the **secldapclntd** daemon. It is also recommended configuration values are specified in the **/etc/security/ldap/ldap.cfg** file instead of using command line flags, so that these values will be used each time you start the **secldapclntd** process.

Related Information

The **mksecldap**, **start-secldapclntd**, **stop-secldapclntd**, **restart-secldapclntd**, **ls-secldapclntd**, and **flush-secldapclntd** commands.

The **/etc/security/ldap/ldap.cfg** file.

secldifconv Command

Purpose

Converts user and group entries of an LDIF from one schema type to another.

Syntax

secldifconv [**-R** *load_module*] **-S** *schematype* **-i** *inputFile* [**-r**]

Description

The **secldifconv** command reads the ldif formatted input file specified by the **-i** option, converts the user and group data using the schema type specified by the **-S** option, and prints the result to stdout. If redirected to a file, the result can be added to an LDAP server with the **ldapadd** command or the **ldif2db** command.

The **-S** option specifies the conversion schema type used for the ldif output. The **secldifconv** command accepts the following schema types:

- **AIX** - AIX schema (aixaccount and aixaccessgroup objectclasses)
- **RFC2307** - RFC 2307 schema (posixaccount, shadowaccount, and posixgroup objectclasses)
- **RFC2307AIX** - RFC 2307 schema with full AIX support (posixaccount, shadowaccount , and posixgroup objectclasses, plus the aixauxaccount and aixauxgroup objectclasses).

The input file specified with the **-i** option can include entries in any of the above supported schemas. The **secldifconv** command will convert user and group entries according to the attribute mapping defined in the **/etc/security/ldap/*.map** files for the corresponding schema type. Only user and group entries will be converted, other entries are output unaltered.

Use of the **-r** option allows the removal of attributes in user and group entries that are not included in the specified output schema. If the option is not specified then unrecognized attributes are assumed to be valid and are output unaltered. Note that if the user or group attribute is defined in the schema **secldifconv** is converting from but not in the schema requested to convert into, then the attribute will not be output. This behavior allows for conversion between the **AIX** and **RFC2307AIX** schemas to the **RFC2307** schema which contains a subset of attributes.

In releases of AIX prior to 5.3, when the **AIX** schema was used to store entries in LDAP, the user's password was stored without an encryption prefix (that is, {crypt}). If the **db2ldif** command is used to generate the input file for **secldifconv**, passwords without an encryption prefix are output in {IMASK} format. In order to convert the {imask} format into the proper {crypt} format, the **-R** option should be used to specify the Loadable I&A module to read the password from for conversions from **AIX** schema type, assuming the system has been previously configured to be an LDAP client.

Care should be taken when adding users and groups from other systems to the LDAP server using the **secldifconv** command output. The **ldapadd** and **ldif2db** commands check only for entry name (user name or group name) but not for the numeric ID when adding entries. Merging users and groups from multiple servers using **secldifconv** output can result in sharing of a numeric ID by multiple accounts, which is a security violation. Note that IBM® Directory Server 5.2 and later supports a unique attribute feature that can be used to avoid this issue.

Flags

-R *load_module*
: Specifies the loadable I&A module used to retrieve the user's password if necessary.

-S *schematype*	Specifies the output LDAP schema type. Valid values are **AIX**, **RFC2307**, and **RFC2307AIX**.
-i *inputFile*	Specifies the input file in ldif format that contains user and group data to convert.
-r	Specifies to remove any attributes that are not defined in the specified schema type.

Exit Status

This command returns the following exit values:

0	The command completed successfully.
>0	An error occurred.
-1	Memory failure (that is, Memory allocation failure).

Examples

1. To convert entries in a ldif formatted file to the `rfc2307` schema, type the following:

   ```
   secldifconv -S rfc2307 -i input.ldif
   ```

   This displays the converted file to stdout in ldif format. User entries and group entries are converted into the **rfc2307** schema type.
2. To convert entries in a ldif formatted file to the `rfc2307aix` schema and remove unrecognized attributes, type the following:

   ```
   secldifconv -R LDAP -S rfc2307aix -i input.ldif -r > convert.ldif
   ```

   This sends the output of the command to the `convert.ldif` file in ldif format. Unrecognized attributes are removed during conversion and user passwords will be requested from the LDAP module if necessary.

Location

/usr/sbin/secldifconv

Files

Mode	File
r	**/etc/security/ldap/2307aixgroup.map**
r	**/etc/security/ldap/2307aixuser.map**
r	**/etc/security/ldap/2307group.map**
r	**/etc/security/ldap/2307user.map**
r	**/etc/security/ldap/aixgroup.map**
r	**/etc/security/ldap/aixuser.map**

Related Information

The db2ldif, ldif2db, and ldapadd commands.

The LDAP Attribute Mapping File Format in *AIX 5L Version 5.3 Files Reference*.

---

sectoldif Command

Purpose

Prints users and groups defined locally to **stdout** in ldif format.

Syntax

sectoldif -d *baseDN* [**-S** *schematype*] [**-u** *username*]

Description

The **sectoldif** command reads users and groups defined locally, and prints the result to **stdout** in ldif format. If redirected to a file, the result can be added to a LDAP server with the **ldapadd** command or the **ldif2db** command.

The **-S** option specifies the schema type used for the ldif output. The **sectoldif** command accepts three schema types:

- **AIX** - AIX schema (**aixaccount** and **aixaccessgroup** objectclasses)
- **RFC2307** - RFC 2307 schema (**posixaccount**, **shadowaccount**, and **posixgroup** objectclasses)
- **RFC2307AIX** - RFC 2307 schema with full AIX support (**posixaccount**, **shadowaccount** , and **posixgroup** objectclasses, plus the **aixauxaccount** and **aixauxgroup** objectclasses).

The **sectoldif** command is called by the **mksecldap** command to export users and groups during LDAP server setup. One needs to be extra cautious when exporting additional users and groups from other systems to the LDAP server using the **sectoldif** output. The **ldapadd** and **ldif2db** commands check only for entry name (user name or group name) but not for the numeric id when adding entries. Exporting users and groups from multiple systems using **sectoldif** output can result in sharing of a numeric id by multiple accounts, which is a security violation.

The **sectoldif** command reads the **/etc/security/ldap/sectoldif.cfg** file to determine what to name the user, group and system sub-trees that the data will be exported to. The **sectoldif** command only exports data to the USER, GROUP and SYSTEM types. The names specified in the file will be used to create sub-trees under the base DN specified with the **-d** flag. Refer to the **/etc/security/ldap/sectoldif.cfg** file documentation for more information.

Flags

-d *baseDN*	Specifies the base DN under which to place the user and group data.
-S *schematype*	Specifies the LDAP schema used to represent user/group entries in the LDAP server. Valid values are AIX, RFC2307, and RFC2307AIX. Default is AIX.
-u *username*	Specifies to print a specific user.

Examples

1. To print all users and groups defined locally, enter the following:

   ```
   sectoldif -d cn=aixsecdb,cn=aixdata -S rfc2307aix
   ```

   This prints all users and groups defined locally to **stdout** in ldif format. User entries and group entries are represented using the rfc2307aix schema type. The base DN is set to `cn=aixsecdb, cn=aixdata`.
2. To print only locally defined user foo, enter the following:

   ```
   sectoldif -d cn=aixsecdb,cn=aixdata -u foo
   ```

   This prints locally defined user foo to **stdout** in ldif format. Without the **-S** option, the default AIX schema type is used to represent foo's ldif output.
3. To export data in a format that is compatible with AIX 4.3 and AIX 5.1 clients, perform the following:
   a. Edit the **/etc/security/ldap/sectoldif.cfg** file to include the following entries:

      ```
      USER            ou      organizationalUnit      aixuser
      GROUP           ou      organizationalUnit      aixgroup
      ID              ou      organizationalUnit      System
      ```

   b. Invoke the **sectoldif** command as follows:

      ```
      sectoldif -d cn=aixsecdb,cn=aixdata -S aix
      ```

This prints all users and groups defined locally to stdout in ldif format. User entries and group entries are represented using the aix schema type. The base DN is set to `cn=aixsecdb,cn=aixdata`, with user subtree `ou=aixuser` and group subtree `ou=aixgroup`.

Files

Mode	File
r	/etc/passwd
r	/etc/group
r	/etc/security/passwd
r	/etc/security/limits
r	/etc/security/user
r	/etc/security/environ
r	/etc/security/user.roles
r	/etc/security/lastlog
r	/etc/security/smitacl.user
r	/etc/security/mac_user
r	/etc/security/group
r	/etc/security/smitacl.group
r	/etc/security/login.cfg

Related Information

The **mksecldap** and **nistoldif** commands.

The **/etc/security/ldap/sectoldif.cfg** file.

---

securetcpip Command

Purpose

Enables the operating system network security feature.

Syntax

securetcpip

Description

The **securetcpip** command provides enhanced security for the network. This command performs the following:

1. Runs the **tcbck -a** command, which disables the nontrusted commands and daemons: **rcp**, **rlogin**, **rlogind**, **rsh**, **rshd**, **tftp**, and **tftpd**. The disabled commands and daemons are not deleted; instead, they are changed to mode 0000. You can enable a particular command or daemon by re-establishing a valid mode.
2. Adds a TCP/IP security stanza to the **/etc/security/sysck.cfg** file. The stanza is in the following format:

```
tcpip:
   netrc = ftp,rexec     /* functions disabling netrc */
```

Before running the **securetcpip** command, acquiesce the system by logging in as root user and executing the **killall** command to stop all network daemons.

> **Attention:** The **killall** command kills all processes except the calling process. If logged in or applications are running, exit or finish before executing the **killall** command.

After issuing the **securetcpip** command, shut down and restart your system. All of your TCP/IP commands and network interfaces should be properly configured after the system restarts.

Files

/etc/security/config	Contains information for the security system.

Related Information

The **killall** command, **tcbck** command.

The **.netrc** file format.

Trusted Processes in *Security*.

---

sed Command

Purpose

A stream editor.

Syntax

sed [**-n**] *Script* [*File* ...]

sed [**-n**] [**-e** *Script*] ... [**-f** *ScriptFile*] ... [*File* ...]

Description

The **sed** command modifies lines from the specified *File* parameter according to an edit script and writes them to standard output. The **sed** command includes many features for selecting lines to be modified and making changes only to the selected lines.

The **sed** command uses two work spaces for holding the line being modified: the pattern space, where the selected line is held; and the hold space, where a line can be stored temporarily.

An edit script consists of individual subcommands, each one on a separate line. The general form of **sed** subcommands is the following:

[*address-range*] *function*[*modifiers*]

The **sed** command processes each input *File* parameter by reading an input line into a pattern space, applying all **sed** subcommands in sequence whose addresses select that line, and writing the pattern space to standard output. It then clears the pattern space and repeats this process for each line specified in the input *File* parameter. Some of the **sed** subcommands use a hold space to save all or part of the pattern space for subsequent retrieval.

When a command includes an address (either a line number or a search pattern), only the addressed line or lines are affected by the command. Otherwise, the command is applied to all lines.

An address is either a decimal line number, a $ (dollar sign), which addresses the last line of input, or a context address. A context address is a regular expression similar to those used in the **ed** command except for the following differences:

- You can select the character delimiter for patterns. The general form of the expression is:
  ```
  \?pattern?
  ```

where ? (question mark) is a selectable character delimiter. You can select any character from the current locale except for the space or new-line character. The \ (backslash) character is required only for the first occurrence of the ? (question mark).

The default form for the pattern is the following:

```
/pattern/
```

A \ (backslash) character is not necessary.

- The **\n** sequence matches a new-line character in the pattern space, except the terminating new-line character.
- A **.** (period) matches any character except a terminating new-line character. That is, unlike the **ed** command, which cannot match a new-line character in the middle of a line, the **sed** command can match a new-line character in the pattern space.

Certain commands called *addressed* commands allow you to specify one line or a range of lines to which the command should be applied. The following rules apply to addressed commands:

- A command line without an address selects every line.
- A command line with one address, expressed in context form, selects each line that matches the address.
- A command line with two addresses separated by commas selects the entire range from the first line that matches the first address through the next line that matches the second. (If the second address is a number less than or equal to the line number first selected, only one line is selected.) Thereafter, the process is repeated, looking again for the first address.

Flags

-e *Script*	Uses the *Script* variable as the editing script. If you are using just one **-e** flag and no **-f** flag, the **-e** flag can be omitted.
-f *ScriptFile*	Uses the *ScriptFile* variable as the source of the edit script. The *ScriptFile* variable is a prepared set of editing commands applied to the *File* parameter.
-n	Suppresses all information normally written to standard output.

Note: You can specify multiple **-e** and **-f** flags. All subcommands are added to the script in the order specified, regardless of their origin.

sed Subcommands

The **sed** command contains the following **sed** script subcommands. The number in parentheses preceding a subcommand indicates the maximum number of permissible addresses for the subcommand.

Notes:

1. The *Text* variable accompanying the **a**, **c**, and **i** subcommands can continue onto more than one line, provided all lines but the last end with a \ (backslash) to quote the new-line character. Backslashes in text are treated like backslashes in the replacement string of an **s** command and can be used to protect initial blanks and tabs against the stripping that is done on every script line. The *RFile* and *WFile* variables must end the command line and must be preceded by exactly one blank. Each *WFile* variable is created before processing begins.
2. The **sed** command can process up to 999 subcommands in a pattern file.

(1) **a** *Text*	Places the *Text* variable in output before reading the next input line.
(2)**b[***label***]**	Branches to the **:** command bearing the *label* variable. If the *label* variable is empty, it branches to the end of the script.
(2)**c**	

Text Deletes the pattern space. With 0 or 1 address or at the end of a 2-address range, places the *Text* variable in output and then starts the next cycle.

(2)**d** Deletes the pattern space and then starts the next cycle.

(2)**D** Deletes the initial segment of the pattern space through the first new-line character and then starts the next cycle.

(2)**g** Replaces the contents of the pattern space with the contents of the hold space.

(2)**G** Appends the contents of the hold space to the pattern space.

(2)**h** Replaces the contents of the hold space with the contents of the pattern space.

(2)**H** Appends the contents of the pattern space to the hold space.

(1)**i**
Text Writes the *Text* variable to standard output before reading the next line into the pattern space.

(2)**l** Writes the pattern space to standard output showing nondisplayable characters as 4-digit hexadecimal values. Long lines are folded.

(2)**l** Writes the pattern space to standard output in a visually unambiguous form. The characters \\\, \\a, \\b, \\f, \\r, \\t, and \\v are written as the corresponding escape sequence. Non-printable characters are written as 1 three-digit octal number (with a preceding backslash character) for each byte in the character (most significant byte first). This format is also used for multibyte characters. This subcommand folds long lines. A backslash followed by a new-line character indicates the point of folding. Folding occurs at the 72nd column position. A $ (dollar sign) marks the end of each line.

(2)**n** Writes the pattern space to standard output if the default output is not suppressed. It replaces the pattern space with the next line of input.

(2)**N** Appends the next line of input to the pattern space with an embedded new-line character (the current line number changes). You can use this to search for patterns that are split onto two lines.

(2)**p** Writes the pattern space to standard output.

(2)**P** Writes the initial segment of the pattern space through the first new-line character to standard output.

(1)**q** Branches to the end of the script. It does not start a new cycle.

(2)**r** *RFile* Reads the contents of the *RFile* variable. It places contents in output before reading the next input line.

(2)**s/***pattern***/***replacement***/***flags*

Substitutes the *replacement* string for the first occurrence of the *pattern* parameter in the pattern space. Any character that is displayed after the **s** subcommand can substitute for the **/** (slash) separator except for the space or new-line character.

See the ″Pattern Matching″ section of the **ed** command.

The value of the *flags* variable must be zero or more of:

g Substitutes all non-overlapping instances of the *pattern* parameter rather than just the first one.

n Substitutes for the *n-th* occurrence only of the *pattern* parameter.

p Writes the pattern space to standard output if a replacement was made.

w *WFile*
Writes the pattern space to the *WFile* variable if a replacement was made. Appends the pattern space to the *WFile* variable. If the *WFile* variable was not already created by a previous write by this **sed** script, the **sed** command creates it.

(2)**t***label* Branches to the **:***label* variable in the script file if any substitutions were made since the most recent reading of an input line execution of a **t** subcommand. If you do not specify the *label* variable, control transfers to the end of the script.

(2)**w***WFile* Appends the pattern space to the *WFile* variable.

(2)**x** Exchanges the contents of the pattern space and the hold space.

(2)**y/***pattern1***/***pattern2***/**

	Replaces all occurrences of characters in the *pattern1* variable with the corresponding *pattern2* characters. The number of characters in the *pattern1* and *pattern2* variables must be equal. The new-line character is represented by **\n**.
(2)**!***sed-cmd*	Applies the specified **sed** subcommand only to lines not selected by the address or addresses.
(0)**:***label*	Marks a branch point to be referenced by the **b** and **t** subcommands. This label can be any sequence of eight or fewer bytes.
(1)**=**	Writes the current line number to standard output as a line.
(2)**{***subcmd* **}**	Groups subcommands enclosed in {} (braces).
(0)	Ignores an empty command.
(0)**#**	If a # (pound sign) appears as the first character on a line of a script file, that entire line is treated as a comment, with one exception. For the first line of a script file only, if the character after the # is an n, the default output will be suppressed. The rest of the line after the #n is ignored.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To perform a global change, enter:

   ```
   sed  "s/happy/enchanted/g" chap1  >chap1.new
   ```

   This command sequence replaces each occurrence of the word `happy` found in the file `chap1` with the word `enchanted`. It puts the edited version in a separate file named `chap1.new`. The **g** character at the end of the **s** subcommand tells the **sed** command to make as many substitutions as possible on each line. Without the **g** character, the **sed** command replaces only the first occurrence of the word `happy` on a line.

   The **sed** command operates as a filter. It reads text from standard input or from the files named on the command line (`chap1` in this example), modifies this text, and writes it to standard output. Unlike most editors, it does not replace the original file. This makes the **sed** command a powerful command when used in pipelines.

2. To use the **sed** command as a filter in a pipeline, enter:

   ```
   pr  chap2 | sed "s/Page *[0-9]*$/(&)/" | enq
   ```

   This command sequence encloses the page numbers in parentheses before printing the file `chap2`. The **pr** command puts a heading and page number at the top of each page, then the **sed** command puts the page numbers in parentheses, and the **enq** command prints the edited listing.

   The **sed** command pattern `/Page *[0-9]*$/` matches page numbers that appear at the end of a line. The **s** subcommand changes this to `(&)`, where the `&` stands for the page number that was matched.

3. To display selected lines of a file, enter:

   ```
   sed  -n  "/food/p" chap3
   ```

   The `sed -n` displays each line in the file `chap3` that contains the word `food`. Normally, the **sed** command copies every line to standard output after it is edited. The **-n** flag stops the **sed** command from doing this. You then use subcommands like **p** to write specific parts of the text. Without the **-n** flag, this example displays all the lines in the file `chap3`, and it shows each line containing `food` twice.

4. To perform complex editing, enter:

   ```
   sed  -f  script.sed  chap4  >chap4.new
   ```

This command sequence creates a **sed** script file when you want to do anything complex. You can then test and modify your script before using it. You can also reuse your script to edit other files. Create the script file with an interactive text editor.

5. A sample **sed** script file:

```
:join
/\\$/{N
s/\\\n//
b join
}
```

This **sed** script joins each line that ends with a \ (backslash) to the line that follows it. First, the pattern `/\\$/` selects a line that ends with a \ for the group of commands enclosed in `{}` (braces). The **N** subcommand then appends the next line, embedding a new-line character. The `s/\\\n//` deletes the \ and embedded new-line character. Finally, `b join` branches back to the label `:join` to check for a \ at the end of the newly joined line. Without the branch, the **sed** command writes the joined line and reads the next one before checking for a second \.

Note: The **N** subcommand causes the **sed** command to stop immediately if there are no more lines of input (that is, if the **N** subcommand reads an end-of-file character). It does not copy the pattern space to standard output before stopping. This means that if the last line of the input ends with a \, it is not copied to the output.

6. To copy an existing file (`oldfile`) to a new file (`newfile`) and replace all occurrences of the `testpattern` text string with the contents of the `$REPL` shell variable, enter:

```
cat oldfile | sed -e "s/testpattern/$REPL/g" > newfile
```

7. To replace all occurrences of A with a, B with b, C with c, and all occurrences of newlines with character Z in the input file, enter:

```
$ sed -f command.file input.file
```

where *command.file* is the script file and *input.file* is the input file.

```
$cat command.file
y/ABC\n/abcZ/
```

Alternatively, the following command can also be executed for the same function:

```
sed "y/ABC\n/abcZ/" input.file
```

Related Information

The **awk** command, **ed** command, **grep** command.

Manipulating Strings with sed in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*

National Language Support in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

---

sedmgr Command

Purpose

Displays and sets Stack Execution Disable flag of the system or executable files.

Syntax

sedmgr [**-m** {**off** | **all** | **select** | **setidfiles**}] [**-o** {**on** | **off**}] [**-c** {**system** | **request** | **exempt**} {*file_name* | *file_group*}] [**-d** {*file_name* | *directory_name*}] [**-h**]

Description

The **sedmgr** command is the manager of the Stack Execution Disable (SED) facility. You can use the command to enable and control the level of stack execution done in the system. This command can also be used to set the various flags in an executable file, controlling the stack execution disable. Any changes to the system wide mode setting will take effect only after a system reboot.

The system wide setting can only be modified by the root user. Other set and reset options on individual executable files will be successful only if the user has write permissions to the file. The SED facility is available only in the AIX 64 bit kernel operating systems.

If invoked without any parameter, the **sedmgr** command will display the current setting in regards to the stack execution disable environment.

For more information, refer to the *Stack Execution Disable Protection* section in Login control in the *Security*.

Flags

-c

Sets or resets the "request" and "exempt" SED flags in the header of an executable file. Also, sets or resets the SED request and exempt checking flag in the headers of all the executable files in a *file_group*. This option requires write privilege to the file, or root privilege if *file_group* is specified.
The possible values are as follows:

system
: If the file has the system flag in the executable's header, the operating system decides the operation for the process based on the system-wide SED flags. When the file does not specify any flags, the operating system also decides the operation for the process based on the system wide SED flags.

exempt
: Sets a flag in the executable's header that indicates that this file does stack/head based execution and as a result needs exemption from the SED mechanism. The SED request checking bit is turned off.

request
: Sets a flag in the executable's header that indicates that this file does not do any stack/data area based execution and as a result is SED capable. The SED exempt checking bit is turned off.

You can specify a file group that represents a group of files, such as TCB files. If the specified file name string does not identify a file, then the string is assumed to identify a *file_group*. Currently only the **TCB_files** file group is defined. You can set or reset the SED request and exempt flags for both 32-bit and 64-bit executables. The **-c** flag cannot be used with the **-m**, **-o**, and **-d** flags.

-d Displays the SED request and exempt checking flag for executable files. The SED request and exempt flags are in the file header of an executable. If a directory is specified, then all executables under that directory and its subdirectories are displayed with their SED related flags. This flag requires read privilege to the *file_name* or *directory_name*. The **-d** flag cannot be used with the **-m**, **-o** and **-c** flags.

-h Displays the syntax of the **sedmgr** command.

-m Sets the system-wide stack execution disable mode if the processor supports SED. Any changes to the system-wide setting require a system reboot to take effect. This option will accept one of the following values:

all Enforces stack execution disable for all files except the ones requesting (marked for) exemption.

off Turns off the stack execution disable functionality on the system.

select Sets the mode of operation to select the set of processes that will be enabled and monitored for stack execution disable. Only processes from files with the "request" SED flag set in their headers will be selected.

setidfiles
Sets the mode of operation so that the operating system performs SED for the files with the "request" SED flag set and enables SED for the executable files with the following characteristics:

- **setuid** files owned by root.
- **setid** files with primary group as "system" or "security".

The configured SED attribute is effective at the next 64-bit kernel boot time. Because the SED attribute in ODM does not affect 32-bit kernels, the SED monitoring flag is turned off in that case. If a processor does not support SED, the **sedmgr** command returns an error with the **-m** flag. The **-m** flag cannot be used with the **-c** and **-d** flags.

-o

This option enables SED to monitor instead of terminating the processes when exceptions occur. This option allows you to evaluate if an executable is doing any legitimate stack execution. This setting works with the system-wide mode set using the **-c** option. The SED Monitoring Control flag is part of the system-wide SED settings stored in ODM. Changing this setting requires root privilege. The possible values for this flag are as follows:

on Turns on the monitoring for SED facility. When operating in this mode, the system will allow the process to continue operating even if an SED related exception occurs. Instead of terminating the process, the operating system logs the exception in the AIX error log subsystem.

off Turns off the monitoring mode for SED facility. In this mode, the operating system terminates any process that violates and raises an exception per SED facility.

The configured SED attribute is effective at the next 64-bit kernel boot time. Because the SED attribute in ODM does not affect 32-bit kernels, the SED monitoring flag is turned off in that case. If a processor does not support SED, the **sedmgr** command returns an error with the **-m** flag. The **-o** flag cannot be used with the **-c** and **-d** flags.

None

If no flag is specified, the **sedmgr** command displays the current setting in regards to the stack execution disable environment. It displays the current SED setting in the kernel **var** structure and the system-wide SED settings in ODM.

Parameters

file_name

Name of the executable file whose SED settings are changed. Requires write privilege.

file_group

Group of executable files whose SED settings are changed when a file name is not specified. Requires root privilege.

directory_name

Directory of executable files and any subdirectories of executable files whose SED checking flags are displayed with the **-d** flag.

Exit Status

0 The command completed successfully.

255 An error occurred.

Security

Access Control: This command should be a standard user command and have the trusted computing base attribute.

Examples

1. To change the system-wide SED Mode flag to **setidfiles** and the SED Control flag to on, type:
   ```
   sedmgr -m setidfiles -o on
   ```
2. To change the SED checking flag to `exempt` for the **plans** file, type:
   ```
   sedmgr -c exempt plans
   ```
3. To change the SED checking flag to **select** for all the executable files marked as a TCB file, type:
   ```
   sedmgr -c request TCB_files
   ```
4. To display the SED checking flag of the **plans** file, type:
   ```
   sedmgr -d plans
   ```

Restrictions

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **sedmgr** command generates the following audit record (event):

Event	Information
SEDMGR_Odm	System wide SED setting.
SEDMGR_File	SED setting in an executable file header.

See Setting up auditing in the Auditing overview section of *Security* for more details about how to properly select and group audit events, and how to configure audit event data collection.

Location

/usr/sbin/sedmgr

Files

/usr/bin/tcbck	Accessed in executable mode.
/usr/bin/ldedit	Accessed in executable mode.

Related Information

The **ldedit** command, “tcbck Command” on page 325.

Auditing overview and Stack Execution Disable (SED) Protection section in Login control in *Security*.

---

send Command

Purpose

Sends a message.

Syntax

send [*File* ... | { **-draft** | **-nodraftfolder** | **-draftfolder +***Folder* | **-draftmessage** *Message* }] [**-alias** *File*] [**-format** | **-noformat**] [**-nomsgid** | **-msgid**] [**-nofilter** | **-filter** *File*] [**-nopush** | **-push**] [**-forward** | **-noforward**] [**-noverbose** | **-verbose**] [**-nowatch** | **-watch**]

Description

The **send** command routes messages through the mail delivery system. If the delivery fails, the **send** command displays an error message. By default, `From:` and `Date:` fields are added to each specified

message. Unless a **$SIGNATURE** environment variable or `signature:` profile entry exists, the **send** command places the sender's address in the `From:` field.

The **send** command puts the current date in the `Date:` field. If the **dist** command calls the **send** command, the **send** command adds `Resent-` to the `From:`, `Date:`, and `Message-ID:` fields.

After successful delivery, the **send** command removes messages from active status by renaming them. The system renames messages by prefacing the current message number with a , (comma). Inactive files are unavailable to the Message Handler (MH) package. However, system commands can still manipulate inactive files. Until you use the **send** command again, you can retrieve an inactive file.

Flags

-alias *File*
Specifies a mail alias file to be searched. Three MH profile entries are required to use MH aliases:

```
ali: -alias Aliases
send: -alias Aliases
whom: -alias Aliases
```

where `Aliases` is the file to be searched. The default alias file is **/etc/mh/MailAliases**.

-draft
Uses the current draft message if no file is specified. Without this flag and when no file is specified, the **send** command asks the user if the current draft message is the one to use.

-draftfolder +*Folder*
Specifies the draft folder that contains the draft message to be sent. The **-draftfolder +***Folder* flag followed by a *Message* parameter is the same as specifying the **-draftmessage** flag.

-draftmessage *Message*
Specifies the message to be sent. You can use one of the following message references as the value of the *Message* parameter:

Number
Number of the message.

cur or . (period)
Current message. This is the default.

first First message in a folder.

last Last message in a folder.

next Message following the current message.

prev Message preceding the current message.

-filter *File*
Uses the format instructions in the specified file to reformat copies of the message sent to the recipients listed in the `Bcc:` field.

-format
Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.

-forward
Adds a failure message to the draft message and returns it to the sender if the **send** command fails to deliver the draft. This flag is the default.

-help
Lists the command syntax, available switches (toggles), and version information.

Note: For MH, the name of this flag must be fully spelled out.

-msgid
Adds a message-identification component (such as `Message-ID:`) to the message.

-nodraftfolder
Undoes the last occurrence of the **-draftfolder +***Folder* flag. This flag is the default.

-nofilter
Removes the `Bcc:` header field from the message for recipients listed in the `To:` and `cc:` fields. The flag then sends the message with minimal headers to recipients listed in the `Bcc:` field. This flag is the default.

-noformat	Prevents alteration of the format of the recipient addresses.
-noforward	Prevents return of the draft message to the sender if delivery fails.
-nomsgid	Prevents addition of a message-identification component. This flag is the default.
-nopush	Runs the **send** command in the foreground. This flag is the default.
-noverbose	Prevents display of information during the delivery of the message to the **sendmail** command. This flag is the default.
-nowatch	Prevents display information during delivery by the **sendmail** command. This flag is the default.
-push	Runs the **send** command in the background. The **send** command does not display error messages on the terminal if delivery fails. Use the **-forward** flag to return messages to you that are not delivered.
-verbose	Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved in sending mail.
-watch	Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved in sending mail.

Profile Entries

The following entries are entered in the *UserMhDirectory*/**.mh_profile** file:

`Draft-Folder:`	Sets the default folder for drafts.
`mailproc:`	Specifies the program used to post failure notices.
`Path:`	Specifies the user's MH directory.
`postproc:`	Specifies the program used to post messages.
`Signature:`	Sets the mail signature.

Examples

To send a draft message that is in your **$HOME/Mail/draft** file, enter:

```
send
```

The system responds with a message similar to the following:

```
Use "/home/david/Mail/draft"?
```

If you enter `yes`, the draft message is sent, and you are returned to the shell prompt. In this example, the name of the **$HOME** directory is **/home/david**.

Files

$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/send	Contains the **send** command.

Related Information

The **ali** command, **comp** command, **dist** command, **forw** command, **post** command, **sendmail** command, **spost** command.

The **.mh_alias** file format, **.mh_profile** file format.

Mail applications in *Networks and communication management*.

sendbug Command

Purpose
Mails a system bug report to a specified address.

Syntax
sendbug [*Address*]

Description
The **sendbug** command is a shell script to assist the user in composing and mailing bug reports in the correct format.

The **sendbug** command starts the editor specified by the **EDITOR** environment variable on a temporary copy of the bug report format outline. The default editor is vi.

Fill out the appropriate fields in the bug report format outline and exit the editor. The **sendbug** command mails the completed report to the address specified by the *Address* parameter. The default address is POSTMASTER.

Files

/usr/lib/bugformat	Contains the bug report outline.

Related Information
The **bugfiler** command, **env** command, **sendmail** command.

Mail management in *Networks and communication management*.

sendmail Command

Purpose
Routes mail for local or network delivery.

Syntax
sendmail [**-ba** | **-bd** | **-bD** | **-bh** | **-bH** | **-bi** | **-bm** | **-bp** | **-bs** | **-bv** | **-bt** [**-C***File*] [**-d***Value*]] [**-B***Type*] [**-F***FullName*] [**-f***Name*] [**-h***Number*] [**-i**] [**-M***x Value*] [**-n**] [**-N** *Dsn*] [**-O***Option*=*Value*] [**-o** *Option* [*Value*]] [**-p***Protocol*] [**-q** [*Time*]] [**-qI***Substr*] [**-qR***Substr*] [**-qS***Substr*] [**-R** *Return*] [**-r** *addr*] [**-t**] [**-U**] [**-V** *Envid*]] [**-v**] [**-X** *LogFile*] *Address*

> **Note:** The *Address* parameter is optional with the **-bd**, **-bi**, **-bp**, **-bt**, and **-q** [*Time*] flags.

Description
> **Note:** On sendmail V8.7 (AIX 4.2 and later), name resolution ordering is DNS, NIS, then local. If you wish to override this specify an **/etc/netsvc.conf** file or NSORDER environment variable.

The **sendmail** command receives formatted text messages and routes the messages to one or more users. Used on a network, the **sendmail** command translates the format of the header information of the message to match the requirements of the destination system. The program determines the network of the destination system by using the syntax and content of the addresses.

The **sendmail** command can deliver messages to:

- Users on the local system
- Users connected to the local system using the TCP/IP protocol
- Users connected to the local system using the Basic Networking Utilities (BNU) command protocol

The **sendmail** command is not intended as a user interface routine; other commands provide user-friendly interfaces. Use the **sendmail** command only to deliver preformatted messages.

The **sendmail** command reads standard input for message text. The **sendmail** command sends a copy of the message to all addresses listed whenever it reads an end of the message character, either an end-of-file (Ctrl-D) control sequence or a single period on a line.

sendmail Mail Filter API (Milter)

The **sendmail** Mail Filter API provides access to mail messages as they are being processed so that third-party programs can filter meta-information and content. Filters that are developed using the **sendmail** Mail Filter API use threads, so it may be necessary to alter the per-process limits in your filter. For example, if your filter will be heavily used, you may want to use the **setrlimit()** subroutine to increase the number of open file descriptors.

Specifying filters in sendmail configs: Filters are specified with the key letter **X** (for external). The following are three example filters:

```
Xfilter1, S=local:/var/run/f1.sock, F=R
Xfilter2, S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m
Xfilter3, S=inet:3333@localhost
```

You can specify filters in your .mc file. The following filter attaches to a UNIX®-domain socket in the /var/run directory:

```
INPUT_MAIL_FILTER(`filter1', `S=local:/var/run/f1.sock, F=R')
```

The following filter uses an IPv6 socket on port 999 of localhost:

```
INPUT_MAIL_FILTER(`filter2', `S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m')
```

The following filter uses an IPv4 socket on port 3333 of localhost:

```
INPUT_MAIL_FILTER(`filter3', `S=inet:3333@localhost')
```

sendmail mail filter flags:

R Reject connection if filter is not available.

T Temporarily fail connection if filter is not available.

If neither F=R or F=T is specified, the message is passed by sendmail as if the filter were not present. The separator is a comma (**,**).

sendmail mail filter timeouts: You can override the default sendmail timeouts with T=*x*. There are four fields in the T= statement:

C Timeout for connecting to a filter (if 0, use system timeout).

S Timeout for sending information from the MTA to a filter.

R Timeout for reading reply from the filter.

E Overall timeout between sending end-of-message to filter and waiting for the final acknowledgment.

The separator between each entry is a semicolon (**;**).

The default values are:

- T=C:0m;S:10s;R:10s;E:5m

The InputMailFilters option determines which filters are invoked and how the filters are sequenced:

```
InputMailFilters=filter1, filter2, filter3
```

This is set automatically according to the order of the INPUT_MAIL_FILTER commands in your .mc file. You can also reset the value by setting confINPUT_MAIL_FILTERS in your .mc file. This option calls the three filters in the order the filters were specified.

You can define a filter without adding it to the input filter list by using **MAIL_FILTER()** instead of **INPUT_MAIL_FILTER()** in your .mc file.

Note: If InputMailFilters is not defined, no filters will be used.

Using the Configuration File

The **sendmail** command uses a configuration file (the **/etc/mail/sendmail.cf** file by default) to set operational parameters and to determine how the command parses addresses. This file is a text file that you can edit with other text editors. After modifying **sendmail.cf**, refresh the **sendmail** daemon.

The current process ID of the **sendmail** command is stored in the **/etc/mail/sendmail.pid** file. Issue the **kill -15** command as follows to have the **sendmail** command reread the newly edited **sendmail.cf**:

```
kill -15 `head -1 /etc/mail/sendmail.pid`
```

If the **srcmstr** command is running, you may issue the **refresh** command, as follows, to build the configuration database, the aliases database, and the NLS database again.

```
refresh -s sendmail
```

The **sendmail** command rereads these databases and continues operation with the new data.

Defining Aliases

The **sendmail** command allows you to define aliases to use when the **sendmail** command handles the local mail. Aliases are alternate names that you can use in place of elaborate network addresses. You can also use aliases to build distribution lists.

Define aliases in the **/etc/mail/aliases** file. This file is a text file you can edit. The **sendmail** command uses a database version of this file. You must build a new alias database by running the **sendmail -bi** command or the **newaliases** command before any changes made to the **/etc/mail/aliases** file become effective.

Berkeley DB support is now available on AIX 5.1 for Sendmail 8.11.0. Sendmail will continue to read the aliases in the DBM format until the aliases database gets rebuilt. Once rebuilt, sendmail will read the aliases in the Berkeley DB format and store them in the **/etc/mail/aliases.db** file.

> **Note:** When defining aliases in the **/etc/mail/aliases** file, use only lowercase characters for nested aliases. Uppercase characters on the right-hand side of an alias are converted to lowercase before being stored in the aliases database. In the following example, mail sent to `testalias` fails, because `TEST` is converted to `test` when the second line is stored.

```
TEST: user@machine
testalias: TEST
```

Every system must have a user or user alias designated as the **postmaster** alias. The default **postmaster** alias is a root file. You can assign this alias to a different user in the **/etc/mail/aliases** file. The

postmaster alias allows other users outside your system to send mail to a known ID and to get information about mailing to users on your system. Also, users on your system can send problem notifications to the **postmaster** ID.

Flags

-B *Type* Sets the body type to *type*. Current legal values are 7BI or 8BITMIME.

Note: The **-b** flag is mutually exclusive.

-ba Starts the **sendmail** command in ARPANET mode. All input lines to the command must end with a carriage return and a line feed (CR-LF). The **sendmail** command generates messages with a CR-LF at the end and looks at the `From:` and `Sender:` fields to find the name of the sender.

-bd Starts the **sendmail** command as a daemon running in the background as a Simple Mail Transfer Protocol (SMTP) mail router.

-bD Starts the **sendmail** command as a daemon running in the foreground as a Simple Mail Transfer Protocol (SMTP) mail router.

-bh Prints the persistent host status database.

-bH Purges the persistent host status database.

-bi Builds the alias database from information defined in the **/etc/mail/aliases** file. Running the **sendmail** command with this flag is the same as running the **/usr/sbin/newaliases** command.

-bm Delivers mail in the usual way. (This is the default.)

-bp Prints a listing of the mail queue. Running the **sendmail** command with this flag is the same as running the **/usr/sbin/mailq** command.

-bs Uses the simple mail transfer protocol (SMTP) as described in RFC821 to collect mail from standard input. This flag also includes all of the operations of the **-ba** flag that are compatible with SMTP.

-bt Starts the **sendmail** command in address test mode. This mode allows you to enter interactive addresses and watch as the **sendmail** command displays the steps it takes to parse the address. At the test-mode prompt, enter a rule set or multiple rule sets separated by commas and an address. Use this mode for debugging the address parsing rules in a new configuration file.

-bv Starts the **sendmail** command with a request to verify the user IDs provided in the *Address* parameter field of the command. The **sendmail** command responds with a message telling which IDs can be resolved to a mailer command. It does not try to collect or deliver a message. Use this mode to validate the format of user IDs, aliases, or mailing lists.

-C*File* Starts the **sendmail** command using an alternate configuration file specified by the *File* variable. Use this flag together with **-bt** to test a new configuration file before installing it as the running configuration file.

-d*Value* Sets the debugging value to the value specified by the *Value* variable. The only valid value is 21.*n*, where *n* is any nonzero integer. This produces information regarding address parsing and is typically used with the **-bt** flag. Higher values of *n* produce more verbose information. Root permissions are required for this flag.

-F*FullName* Sets the full name of the sender to the string provided in the *FullName* variable.

-f*Name* Sets the name of the from person (the envelope sender of the mail). This address may also be used in the From: header if that header is missing during initial submission. The envelope sender address is used as the recipient for delivery status notifications and may also appear in a Return-path: header. This flag should only be used by trusted users (normally root, daemon, and uucp) or if the person you are trying to become is the same as the person you are. Otherwise, an X-Authentication-Warning header is added to the message.

-h*Number* Sets the hop count to the value specified by the *Number* variable. The hop count is the number of times that the message has been processed by an SMTP router (not just the local copy of the **sendmail** command). The mail router increments the hop count every time the message is processed. When it reaches a limit, the message is returned with an error message in order to prevent infinite loops in the mail system.

-i Ignores dots alone on lines by themselves in incoming messages. This should be set if you are reading data from a file.

-L Sets the identifier used in syslog messages to the supplied tag.

-M*x Value*	Sets marco *x* to the specified *value*.
-N *Dsn*	Sets delivery status notification conditions to DSN. The delivery status notification conditions can be: `never` for no notifications or for a comma separated list of the values, `failure` for notification if delivery failed, `delay` for notification if delivery is delayed, and `success` for notification when the message is successfully delivered.
-n	Prevents the **sendmail** command from interpreting aliases.
-O *Option*=*Value*	Sets *Option* to the specified *Value*. Use for long-form option names. This flag applies only to AIX 4.2 or later.
-o*Option* [*Value*]	Sets the *Option* variable. If the option is a valued option, you must also specify a value for the *Value* variable. **Note:** For valid values, see ″Options for the sendmail Command in the sendmail.cf File″ in *Networks and communication management*.
-p*Protocol*	Sets the sending protocol. It is recommended that you set this. You can set *Protocol* in the form *Protocol*:*Host* to set both the sending protocol and the sending host. For example, `-pUUCP:uunet` sets the sending protocol to UUCP and the sending host to uunet. Some existing programs use **-oM** flag to set the **r** and **s** macros, which is equivalent to using the **-p** flag. This flag applies only to 4.2 or later.
-ql*Substr*	Limits process jobs to those containing *Substr* as a substring of the queue ID.
-qR*Substr*	Limits process jobs to those containing *Substr* as a substring of one of the recipients.
-qS*Substr*	Limits process jobs to those containing *Substr* as a substring of the sender.
-q [*Time*]	Processes saved messages in the queue at the intervals specified by the *Time* variable. If the *Time* variable is not specified, this flag processes the queue at once.
-R *Return*	Sets the amount of the message to be returned if the message bounces. The *Return* parameter can be `full` to return the entire message or `hdrs` to return only the headers.
-r *addr*	An obsolete form of **-f**.
-t	Sends the message to the recipients specified in the `To:`, `Cc:`, and `Bcc:` fields of the message header, as well as to any users specified on the command line.
-U	Sets initial (user) submission. This should always be set when called from a user agent such as Mail or exmh and should never be set when called by a network delivery agent such as rmail.
-V *Envid*	Sets the original envelope ID. This is propagated across SMTP to servers that support DSNs and is returned in DSN-compliant error messages.
-v	Starts the **sendmail** command in verbose mode. The **sendmail** command displays messages regarding the status of transmission and the expansion of aliases.
-X *LogFile*	Logs all traffic in and out of **sendmail** in *LogFile* for debugging mailer problems. Use this flag sparingly, since it produces a lot of data very quickly. This flag applies only to 4.2 or later.

You can also set or remove the **sendmail** configuration processing options. The person responsible for the mail system uses these options. To set these options, use the **-o** flag on the command line or the **O** control line in the configuration (**/etc/mail/sendmail.cf**) file.

Example

Run the following command to display the sendmail version:

```
echo \$Z | sendmail -d0
```

The system responds with a message similar to the following:

```
Version AIX5.2/8.11.6p2
 Compiled with: LDAPMAP MAP_REGEX LOG MATCHGECOS MIME7TO8 MIME8TO7
                NAMED_BIND NDBM NETINET NETINET6 NETUNIX NEWDB NIS NISPLUS
                QUEUE SCANF SMTP USERDB XDEBUG

============ SYSTEM IDENTITY (after readcf) ============
      (short domain name) $w = dodgers
  (canonical domain name) $j = dodgers.usca.ibm.com
         (subdomain name) $m = usca.ibm.com
              (node name) $k = dodgers
```

```
=====================================================

Recipient names must be specified
# oslevel -r
5200-02
#
```

Security

Auditing Events:

Event	Information
SENDMAIL_Config	Configuration event
SENDMAIL_ToFile	File-creation event

Exit Status

The **sendmail** command returns exit status values. These exit values are defined in the **/usr/include/sysexits.h** file. The following table summarizes the meanings of these return values:

EX_CANTCREAT	The **sendmail** command cannot create a file that the user specified.
EX_CONFIG	An error was found in the format of the configuration file.
EX_DATAERR	The input data was incorrect in some way.
EX_IOERR	An error occurred during I/O.
EX_NOHOST	The **sendmail** command could not recognize the specified host name.
EX_NOINPUT	An input file (not a system file) did not exist or was not readable.
EX_NOPERM	The user does not have permission to perform the requested operation.
EX_NOUSER	The **sendmail** command could not recognize a specified user ID.
EX_OK	The **sendmail** command successfully completed.
EX_OSERR	A temporary operating system error occurred. An example of such an error is a failure to create a new process.
EX_OSFILE	A system file error occurred. For example, a system file (such as **/etc/passwd**) does not exist, cannot be opened, or has another type of error preventing it from being used.
EX_PROTOCOL	The remote system returned something that was incorrect during a protocol exchange.
EX_SOFTWARE	An internal software error occurred (including bad arguments).
EX_TEMPFAIL	The **sendmail** command could not create a connection to a remote system. Try the request again later.
EX_UNAVAILABLE	A service or resource that the **sendmail** command needed was not available.
EX_USAGE	The command syntax was not correct.

Files

/usr/sbin/sendmail	Contains the **sendmail** command.
/usr/sbin/mailq	Contains the mail queue.
/usr/sbin/newaliases	Contains the alias database.
/usr/sbin/mailstats	Contains statistics found in the **/usr/lib/sendmail.st** file.
/etc/mail/aliases	Contains the text version of the **sendmail** command aliases.
/etc/mail/aliases.db	Contains Berkeley DB formatted database for aliases.
/etc/mail/aliases.dir	Contains DBM formatted database for aliases.
/etc/mail/aliases.pag	Contains DBM formatted database for aliases.
/etc/mail/sendmail.cf	Contains the text version of the **sendmail** configuration file.
/etc/sendmail.st	Contains mail routing statistics information.
/usr/lib/smdemon.cleanu	Maintains aging copies of the log file found in the **/var/spool/mqueue** directory.
/var/spool/mqueue	Contains the temporary files and the log file associated with the messages in the mail queue.
/usr/bin/uux	Contains the mailer command to deliver Basic Networking Utilities (BNU) mail.
/usr/bin/bellmail	Contains the mailer command to deliver local mail.

Related Information

The **bellmail** command, **kill** command, **mail, Mail** command, **mailq** command, **mailstats** command, **newaliases** command, **refresh** command, **uux** command.

The **srcmstr** daemon.

BNU Overview, Mail management, and Transmission Control Protocol/Internet Protocol in *Networks and communication management*.

Mail applications in *Networks and communication management*.

---

setclock Command

Purpose

Sets the time and date for a host on a network.

Syntax

/usr/sbin/setclock [*TimeServer*]

Description

The **/usr/sbin/setclock** command gets the time from a network time server, and if run by a user with root user authority, sets the local time and date accordingly.

The **setclock** command takes the first response from the time server, converts the calendar clock reading found there, and displays the local date and time. If the **setclock** command is run by the root user, it calls the standard workstation entry points to set the system date and time.

If no time server responds or if the network is not operational, the **setclock** command displays a message to that effect and leaves the current date and time settings of the system unchanged.

> **Note:** Any host running the **inetd** daemon can act as a time server.

Parameter

TimeServer — The host name or address of a network host that services TIME requests. The **setclock** command sends an Internet TIME service request to a time server host. If the *TimeServer* name is omitted, the **setclock** command sends the request to the default time server. The default time server in a DOMAIN environment is specified by the name server. Otherwise the default time server is specified in the **/etc/hosts** file.

Examples

1. To display the date and time using the time server host specified in the **/etc/hosts** file, enter:

   ```
   setclock
   Sat Mar 11 15:31:05 1988
   ```

   The **setclock** command displays the proper date and time.
2. To set the date and time, enter:

   ```
   su root
   setclock host1
   Thu Jan 12 15:24:15 1990
   ```

You must use the **su** command or log in as the root user before setting the time from the time server in `host1`.

Related Information

The **timedc** command.

The **inetd** daemon, **timed** daemon.

The **hosts** file format.

TCP/IP daemons in *Networks and communication management*.

---

setea Command

Purpose

Writes or deletes a named extended attribute to a file.

Syntax

setea -n *Name* [**-l**]{ **-v** *Value* | **-d** | **-f** *EAFile* } *FileName* ...

Description

The **setea** command writes or deletes a named extended attribute to a file. The file must be in a file system which supports named extended attributes, such as JFS2 using **v2** extended attribute format.

Note: To prevent naming collisions, JFS2 has reserved the 8-character prefix `(0xf8)SYSTEM(0xF8)` for system-defined extended attributes. Avoid using this prefix for naming user-defined extended attributes.

This command is not used to set ACLs. To set ACLs, use the **aclput** command.

Flags

-d	Specifies to delete the named extended attribute from the file.
-f *EAFile*	*EAFile* specifies a file which contains the EA value. If an extended attribute matching the specified name already exists for the *FileName*, then the value will be changed to the value specified.
-l	Specifies to write or delete the extended attribute from the symbolic link itself rather than the file to which it is pointing.
-n *Name*	Specifies the name of the extended attribute to be written.
-v *Value*	Specifies the value of the named extended attribute. If an extended attribute matching the specified name already exists for the file, then the value will be changed to the value specified. Value is treated as a character string. It should be enclosed in quotes if it contains spaces.
FileName ...	Specifies the file or files from which to write or delete the extended attribute.

Exit Status

0	Successful completion.
Positive integer	An error occurred.

Examples

1. To create an extended attribute with a name of `Approver` and a value of `Grover` for file `design.html`.
   ```
   setea -n Approver -v Grover design.html
   ```
2. To modify an extended attribute named `Approver` to new value of `Joon` for file `design.html`.
   ```
   setea -n Approver -v Joon design.html
   ```
3. To remove an extended attribute named `Approver` from file `design.html`.
   ```
   setea -n Approver -d design.html
   ```
4. To create an extended attribute with a name of `Approver` and a value of `Zach` for the symbolic link `design.html`, type:
   ```
   setea -n Approver -v Zach -l design.html
   ```

Location

/usr/sbin

Related Information

chfs and **crfs** command in *AIX 5L Version 5.3 Commands Reference, Volume 1*.

getea command in *AIX 5L Version 5.3 Commands Reference, Volume 2*.

---

setgroups Command

Purpose

Resets a session's process group set.

Syntax

setgroups [**-**] [**-a** *GroupSet*] [**-d** *GroupSet*] [**-r** [*Group*]] [*GroupSet*]

Description

The **setgroups** command, by default, displays the user's current group set and process group set for the current shell. A user's group set is defined in the user database files. When given a flag and a *GroupSet* parameter, this command resets the process group set as listed by the *GroupSet* parameter. The *GroupSet* parameter is a comma-separated list of group names. The available groups are defined in the user database files.

You can also use the **setgroups** command to add or delete groups from the current group set. Using the **-r** flag, you can reset the real group ID. If you specify the *Groupset* parameter but no flags, the **setgroups** command resets all the groups and makes the first group in the list the real group. The **setgroups** command does not change the security characteristics of the controlling terminal.

When you run the **setgroups** command, the system always replaces your shell with a new one. The command replaces your shell regardless of whether the command is successful or not. For this reason, the command does not return error codes.

The **setgroups -r** command is identical to the **newgrp** command.

Flags

-a *GroupSet* — Adds the groups specified by the *GroupSet* parameter to the current session. The number of groups in the new set must not exceed **NGROUPS_MAX** groups, a value defined in the **limits.h** file. The real group ID is not changed.

-d *GroupSet*	Removes the groups specified by the *GroupSet* parameter from the current session. If the real group is removed, the next group listed in the current set becomes the real group.
-r *Group*	Resets the real group for the current process. If you do not specify a *Group* parameter and the current real group is not the primary group, the **-r** flag removes the current real group and resets the real group to the original primary group. If you specify a *Group* parameter, this behaves identically to the **newgrp** command.
-	Re-initializes the group set of the session to its original login state.

Security

Access Control: This command should be a general user program. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	Files
r	**/etc/passwd**
r	**/etc/group**

Auditing Events:

Event	Information
USER_SetGroups	realgroup, groupset

Examples

1. As user `sah`, you can display your current group membership and process group set, by entering:

   ```
   setgroups
   ```

   Output similar to the following appears:

   ```
   sah:

      user groups = staff,payroll
      process groups = staff,payroll
   ```

2. To add the `finance` group to the process group of the current session, enter:

   ```
   setgroups  -a  finance
   ```

3. To set your real group to `finance`, enter:

   ```
   setgroups finance,staff,payroll
   ```

   This sets `finance` as the real group. The `staff` and `payroll` groups make up the supplementary group list.

4. To delete the `payroll` group from the current process group set, enter:

   ```
   setgroups  -d  payroll
   ```

5. To change the process group set back to your default set, enter:

   ```
   setgroups   -
   ```

   This resets your current session to its original state just after you log in.

Files

/usr/bin/setgroups	Contains the **setgroups** command.
/etc/group	Contains basic group attributes.
/etc/passwd	Contains basic user attributes.

Related Information

The **login** command, **newgrp** command, **setsenv** command, **tsm** command.

---

setmaps Command

Purpose

Sets terminal maps or code set maps.

Syntax

To use setmaps with no input or output map file designation, type the following:

setmaps [**-v**] [**-c** | **-h**]

To select a file from the default directory as the code set map file, type the following:

setmaps [**-v**] **-s** **-i** *MapName*

To select a designated file as the code set map file, type the following:

setmaps [**-v**] **-s** **-I** *File1*

To select a file from the default directory as the input or output terminal map file, type the following:

setmaps [**-v**] [**-D**] [**-k** *KeyName*] [**-d** *DirectoryPath*] { **-i** | **-o** } *MapName*

To select files from the default directory as the input or output terminal map files, type the following:

setmaps [**-v**] [**-D**] [**-d** *DirectoryPath*] **-t** *MapName*

To select a designated file as the input or output terminal map file, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] { **-I** | **-O** } *File1*

To load the default terminal map file for later use, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] [**-r**] **-l** *File2*

To load a designated terminal map file for later use, type the following:

setmaps [**-v**] [**-D**] [**-k** KeyName] [**-r**] **-L** *File1*

Description

Note: If this command is run without root user authority, the code set map is not loaded, only debugged.

The **setmaps** command handles terminal and code set maps. The **-s** flag must be used for code set maps. The operating system uses input and output terminal maps to convert internal data representations to the ASCII characters supported by asynchronous terminals. If you enter the **setmaps** command with no flags, it displays the names of the current input and output terminal maps.

A terminal map is a text file containing a list of rules that associate a pattern string with a replacement string. This file normally resides in the **/usr/lib/nls/termmap** directory. The operating system uses an input map file to map input from the keyboard to an application and an output map file to map output from an application to the display.

Terminal mapping works as follows:

1. The system collects characters in a buffer until a pattern specified by a rule in the map file matches a substring in the buffer.
2. The system then constructs and returns the replacement string specified by the rule.

This processing continues with the remaining characters in the buffer.

The rules of a terminal map can test and change the state of the pattern processor. The state is identified by a single-byte character, conventionally a digit (0 through 9). The state is reset to 0, the initial state, whenever the system loads a new map or flushes the terminal input or output buffer (such as when it processes a KILL or INTR character or when a program issues an **ioctl** system call). A terminal map can use states to detect multibyte escape sequences, among other tasks. You can test for state *x* by specifying @x in a pattern. You can set the state to *x* by including @x in the replacement string.

The **setmaps** command, when using the **-s** flag, assigns a code set map to the standard input device. The operating system uses code set maps to determine the number of bytes of memory a character requires and the number of display columns it requires.

Flags

-c Clears all mappings on this terminal.

-d *DirectoryPath* Causes the *DirectoryPath* variable to be used as the path to the directory that contains the *MapName* variable. Specifying this flag and variable overrides the **/usr/lib/nls/termmap** directory.

-D Produces a debug program printout of the specified map on the standard output device before loading the map. When using this to run the debug program on new maps, do not run with root user authority until the map is fully debugged to prevent the map from actually being loaded.

-h Prints the usage information of the **setmaps** command (used with the **-v** flag for advanced users).

-i *MapName* Selects the **/usr/lib/nls/termmap/MapName.in** file as the input map. When used with the **-s** flag, this flag selects the **/usr/lib/nls/csmap/MapName** file as the terminal code set map file.

-l *File1* Selects the contents of the *File1* variable as the input map. The file specified by the *File1* variable can be either a full path name or a path name relative to the current working directory. When used with the **-s** flag, this flag selects the contents of the *File1* variable as the terminal code page map file.

-k *KeyName* Associates the contents of the *KeyName* variable with the map being selected. This key name overrides the default key, which is normally set to the value of the *MapName* variable.

-l *File2* Loads the **/usr/lib/nls/termmap/File2** file for later use. The *File2* variable includes the full path name and suffix (if any) of the map file.

Note: You must have root user authority to specify this flag.

-L *File1* Loads the specified map for later use. The *File1* variable includes the full path name and suffix (if any) of the map file.

Note: You must have root user authority to specify this flag.

-o *MapName* Selects the **/usr/lib/nls/termmap/MapName.out** file as the terminal output map.

-O *File1* Selects the contents of the *File1* variable as the terminal output map. The *File1* variable includes the full path name and suffix (if any) of the map file.

-r Forces reloading of the specified map, even if it is already loaded. Terminals using the old map continue to do so until they are logged off or until their maps are explicitly reset. If you do not specify this flag, a map is loaded only if it has not already been loaded into the kernel.

Note: You must have root user authority to specify this flag.

-s Treats any map as a code set map.

-t *MapName* Selects the **/usr/lib/nls/termmap/MapName.in** file as the terminal input map and the **/usr/lib/nls/termmap/MapName.out** file as the terminal output map.

-v Selects verbose output.

All maps loaded must have unique names. Use the **-k** flag to eliminate naming conflicts. Only the **-i**, **-o**, and **-t** flags implicitly add a suffix. Other flags specifying map names should include a suffix if appropriate. If a requested map name is already loaded in the kernel, that map is used even if the path information provided on the command line implies a different map.

To reset the code set map to its original state, the **/usr/lib/nls/csmap/sbcs** code set map should be used.

Examples

1. To display the current map settings for this terminal, enter:
   ```
   setmaps
   ```
2. To clear all mapping for the current terminal, enter:
   ```
   setmaps -c
   ```
3. To set up mapping (both input and output maps) for an `ibm3161-C` terminal, enter:
   ```
   setmaps -t ibm3161-C
   ```
4. To load the `vt220` input map into the kernel as the `fred` map, enter:
   ```
   setmaps -k fred -i vt220
   ```
5. To gather debug output for a new map called `bob` in a file called `bob.dump`, enter:
   ```
   setmaps -D -L /tmp/bob > bob.dump
   ```
6. To set up a code set map conforming to the IBM-932 code page for this terminal, enter:
   ```
   setmaps -s -i IBM-932
   ```
7. To set up a code set map conforming to the IBM-943 code page for this terminal, enter:
   ```
   setmaps -s -i IBM-943
   ```
8. To set up a code set map from the file `myEUC` for this terminal, enter:
   ```
   setmaps -s -I myEUC
   ```

Files

/usr/bin/setmaps Contains the **setmaps** command.

/usr/lib/nls/termmap/*.in Contains input map files.

/usr/lib/nls/termmap/*.out Contains output map files.

/usr/lib/nls/csmap/sbcs Contains code set map for a single-byte code page.

/usr/lib/nls/csmap/IBM-943 Contains code set map for the IBM-943 code page.

/usr/lib/nls/csmap/IBM-eucJP Contains code set map for the IBM-eucJP code page.

Related Information

The **stty** command.

The **setmaps** file format, **termios.h** file.

The **setcsmap** subroutine.

National Language Support in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

---

setsenv Command

Purpose

Resets the protected state environment of a user.

Syntax

setsenv [**-**] *NewEnvironment*

Description

The **setsenv** command resets your protected state environment while you are logged in. The protected state environment is defined as a set of variables. These variables are kept in the kernel and can be modified only by a **SETUINFO** system call. The **setsenv** command uses the variables specified by the *NewEnvironment* parameter. This parameter consists of *EnvironmentVariable*=*Value* definitions separated by a blank space. For information on environment variables, see **environment** File.

You cannot reset the following environment variables with the **setsenv** command:

NAME Your last authenticated user name. This corresponds to the real user ID of the current process.

TTY The name of the terminal on which you logged in. This corresponds to the initial controlling terminal for the process. This variable cannot be set for processes initiated without a *full login*. A full login is a login initiated by the **getty** command.

LOGNAME The name under which you logged in, if the current session was started from a terminal login program. If the session was not started from a terminal, this variable is not set.

If you enter the **setsenv** command without any defined variables, it displays the current protected state. The **setsenv** command does not change the security characteristics of the controlling terminal.

When you run the **setsenv** command, it replaces your current shell and gives you a new one. The command replaces your shell regardless of whether it completed successfully or not. For this reason, the command does not return error codes.

Flags

- Reinitializes the environment as if the user had just logged in to the system. Otherwise, the environment is not changed.

Security

Access Control: This command should be a standard user program. This command should be installed as a program in the trusted computing base (TCB). The command should be owned by the root user with the **setuid** (SUID) bit set.

Files Accessed:

Mode	File
r	/etc/environment
r	/etc/security/environ

Auditing Events:

Event	Information
USER_SetEnv	new environment string

Examples

1. To display the current environment variables, enter:
   ```
   setsenv
   ```
2. To add the `PSEUDO=tom` protected environment variable, enter:
   ```
   setsenv PSEUDO=tom
   ```

   This example sets a user name for the **PSEUDO** protected environment variable.

Files

/usr/bin/setsenv	Specifies the path to the **setsenv** command.
/etc/environment	Contains environment information for each user.
/etc/security/environ	Contains privileged environment information for each user.

Related Information

The **login** command, **setgroups** command, **su** command, **tsm** command.

The **getuinfo** subroutine, **setpenv** subroutine, **usrinfo** subroutine.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to Securing the network in *Security*.

---

settime Command

Purpose

Updates access and modification times of a file.

Syntax

settime [[*MMddhhmm*[*yy*]] | [**-f** *ReferenceFile*]] *File* ...

Description

settime updates the argument files with the current access and modification times by default. The file is not created if it does not exist. The **settime** command silently continues its operation if the file does not exist.

Note: Any dates beyond and including the year 2038 are not valid for the **settime** command.

Flags

-f *ReferenceFile*	Use the corresponding time of *ReferenceFile* instead of the current time.

Parameters

MMddhhmm[*yy*]	Time is specified for the **settime** command in the format *MMddhhmm* or *MMddhhmmyy*, where *MM* is a two-digit representation of the month, *dd* is a two-digit representation of the day of the month, *hh* is a two-digit representation of the hour, *mm* is a two-digit representation of the minute, and *yy* is a two-digit representation of the year.
File	Specifies the name of a file or a space separated list of files.

Exit Status

0 The command completed successfully.

>0 An error occurred.

The return code from **settime** is the number of specified files for which the times could not be successfully modified.

Examples

1. To update the access and modification times of the file ″infile″ to the current time, enter:
   ```
   settime infile
   ```
2. To update the access and modification times of ″infile″ to be the same as ″reffile″, enter:
   ```
   settime -f reffile infile
   ```
3. To update the access and modification times of multiple files, enter:
   ```
   settime file1 file2 file3
   ```
4. To update the access and modification times of a file to April 9th 2002 with time 23:59, enter:
   ```
   settime 0409235902 infile
   ```

Files

/usr/bin/settime	Contains the **settime** command.

Related Information

The **touch** command.

---

setuname Command

Purpose

Sets the node name of the system.

Syntax

setuname [**-t**] **-n** *Node*

Description

The **setuname** command is used to set the node name of the system. The **-n** option must be specified. Only users with root authority can set the node name. The change can be made temporary by using the **-t** option. The node name will be modified only on the current running kernel if a temporary change is requested. The nodename set temporarily will not persist after a reboot. Without the **-t** option the node name is changed permanently in the ODM database.

Flags

-n *Node*	Specifies that the node name has to be changed. This option is required. *Node* is the primary node name for the host. This can be the UUCP communications network name for the system.
-t	Temporary change. No attempt will be made to make the change permanent. The original name will be restored after reboot.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To temporarily change the node name to ″orion″, enter:
   ```
   setuname -t -n orion
   ```
2. To permanently change the node name to ″orion″, enter:
   ```
   setuname -n orion
   ```

Files

/usr/bin/setuname	Contains the setuname command.

Related Information

The **chdev** command, **hostname** command, **uname** command.

---

sh Command

Purpose

Invokes the default shell.

Syntax

Refer to the syntax of the **ksh** command. The **/usr/bin/sh** file is linked to the Korn shell.

Description

The **sh** command invokes the default shell and uses its syntax and flags. The shell linked to the **/usr/bin/sh** path is the default shell. The standard configuration of the operating system links the **/usr/bin/sh** path to the Korn shell. See Korn shell or POSIX shell built-in commands in *Operating system and device management* for specific information about Korn shell features.

Flags

Refer to the flags for the Korn shell (**ksh** command).

Files

/usr/bin/sh	Contains the **sh** command.

Related Information

The **ksh** command.

Korn shell or POSIX shell built-in commands, Shells in *Operating system and device management*.

---

shconf Command

Purpose

Manages the system hang detection parameters.

Syntax

shconf -d

shconf -R -l *Name*

shconf {**-D** [**-O**] | **-E** [**-O**]} [**-H**] **-l** *Name*

shconf -l *Name* [**-a** *Attribute=Value*] ...

Description

The **shconf** command is used to display or specify the parameters of the priority problem detection and lost I/O detection.

For the priority problem, the user can specify five actions described below and for each action, the user can specify the priority level to check, the time out while no process or thread executes at a lower or equal priority, the terminal device for the warning action, and the getty action:

pp_cmd	Launches a command specified by the path parameter.
pp_errlog	Logs an error in error log.
pp_login	Launches a getty at the highest priority on the serial line specified by the terminal device parameter (term).
pp_reboot	Reboots the system.
pp_warning	Displays a warning message on the console specified by the terminal device parameter (term).

For lost I/O, the user can specify the actions listed below and **errlog**, which is automatic when lost I/O detection is enabled. There is a unique timeout which applies to all enabled actions.

lio_warning	Displays a warning message on the console specified by the terminal device parameter (term).
lio_reboot	Creates a system dump and reboots the system

Flags

-d	Displays if priority problem detection and lost I/O detection are enabled or not.
-R	Restore the default values for a specified name of detection.
-a *Attribute=Value*	Specifies the attribute value pairs used for changing specific attribute values.
-D	Displays the default values for a specified name of detection.
-E	Displays the effective values for a specified name of detection.
-H	Displays the headers above the column output. When used together, the **-O** flag overrides the **-H** flag.
-I *Name*	Specifies the detection name.
-O	Displays all attribute names separated by colons and, on the second line, displays all the corresponding attribute values separated by colons. The attribute values are current values when the **-E** flag is also specified and default values when the **-D** flag is specified. This flag cannot be used with the **-a** flag.

Files

/usr/sbin/shconf	Contains the **shconf** command.

shell Command

Purpose

Executes a shell with the user's default credentials and environment.

Syntax

shell

Description

The **shell** command re-initializes a user's login session. When the command is given, the port characteristics of the process's controlling terminal are reset and all access to the port is revoked. The **shell** command then resets the process credentials and environment to the defaults established for the user and executes the user's initial program. All credentials and environment are established according to the login user ID of the invoking process.

If the **shell** command is invoked on the trusted path and the user's **tpath** attribute in the **/etc/security/user** file does not have a value of **always**, the trusted environment of the terminal is not maintained.

> **Note:** The **shell** command does not reset the login ID of the user.

Security

Access Control: The command should be **setuid** to the root user to reset the user's process credentials, and grant execute (x) access to all users. The command should have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/group**
r	**/etc/security/audit/config**
r	**/etc/security/environ**
r	**/etc/security/limits**
r	**/etc/security/user**

Auditing Events:

Event	Information
USER_Shell	portname

Examples

To re-initialize your session to your default credentials and environment after using the trusted shell (**tsh**), enter:

```
shell
```

Files

/usr/bin/shell	Contains the **shell** command.
/etc/security/user	Contains the extended attributes of users.
/etc/passwd	Contains user IDs.
/etc/group	Contains group IDs.
/etc/security/audit/config	Contains the audit configuration information.
/etc/security/environ	Defines the environment attributes for users.
/etc/security/limits	Defines process resource limits for each user.

Related Information

The **getty** command, **init** command, **login** command, **logout** command, **setgroups** command, **su** command, **tsh** command, **tsm** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to Securing the network in *Security*.

---

show Command

Purpose

Shows messages.

Syntax

show [**+***Folder*] [**-draft** | *Messages*] [**-header** | **-noheader**] [**-showproc** *CommandString* | **-noshowproc**]

Description

The **show** command displays the contents of messages. If standard output is not a display, the **show** command lists each message with a one-line header and two separation lines. By default, the **show** command displays the current message in the current folder.

The **show** command invokes a listing program to create the list. The default listing program is **/usr/bin/more**. You can define your own default with the `showproc:` entry in your **$HOME/.mh_profile** file. If you set the `showproc:` entry to `mhl`, the **show** command calls an internal **mhl** routine instead of the **mhl** command. You can also specify the program to perform a listing in the *CommandString* parameter of the **-showproc** flag.

The **show** command passes any flags it does not recognize to the listing program. Thus, you can specify flags for the listing program, as well as for the **show** command.

If the `Unseen-Sequence:` entry is present in your **$HOME/.mh_profile** file and the entry is not empty, the **show** command removes each of the messages shown from each sequence named by the profile entry. If several messages are specified, the last message shown becomes the current message.

Flags

-draft
Shows the *UserMhDirectory*/**draft** file if it exists.

+*Folder*
Specifies a folder. The current folder is the default.

-header
Displays a one-line description of the message being shown. The description includes the folder name and message number. If you show more than one message, this flag does not produce message headers. The **-header** flag is the default.

-help
Lists the command syntax, available switches (toggles), and version information.

Note: For MH, the name of this flag must be fully spelled out.

Messages
Specifies the messages to show. You can specify several messages, a range of messages, or a single message. Use the following references to specify messages:

Number
Number of the message.

Sequence
A group of messages specified by the user. Recognized values include:

all All messages in a folder.

cur or . (period)
Current message. This is the default.

first First message in a folder.

last Last message in a folder.

next Message following the current message.

prev Message preceding the current message.

-noheader
Prevents display of a one-line description of each message.

-noshowproc
Uses the **/usr/bin/cat** command to perform the listing. This is the default.

-showproc *CommandString*
Uses the specified command string to perform the listing.

Profile Entries

The following entries are entered in the *UserMhDirectory*/**.mh_profile** file:

`Current-Folder:` Sets the default current folder.
`Path:` Specifies the user's MH directory.

`showproc:`	Specifies the program used to show messages.
`Unseen-Sequence:`	Specifies the sequences used to keep track of the unseen messages.

Examples

1. To display the contents of the current message in the current folder one screen at a time, enter:

   ```
   show
   ```

   If the message continues for more than one screen, press the Enter key until you have read the entire message.
2. To see the contents of all the messages in the current folder, enter:

   ```
   show all
   ```

   If the messages continue for more than one screen, press the Enter key until you have read all the messages.
3. To see the contents of message `5` in the `meetings` folder, enter:

   ```
   show  +meetings 5
   ```
4. To see the contents of all the messages belonging to the `weekly` sequence in the `meeting` folder, enter:

   ```
   show  +meeting weekly
   ```

Files

$HOME/.mh_profile	Specifies the MH user profile.
UserMhDirectory/**draft**	Contains the current message draft.
/usr/bin/show	Contains the **show** command.

Related Information

The **mhl** command, **next** command, **pick** command, **prev** command, **scan** command, **sendmail** command.

The **.mh_alias** file format, **.mh_profile** file format.

Mail applications in *Networks and communication management*.

---

showmount Command

Purpose

Displays a list of all clients that have remotely mounted file systems.

Syntax

/usr/bin/showmount [**-a**] [**-d**] [**-e**] [*Host*]

Description

The **showmount** command displays a list of all clients that have remotely mounted a file system from a specified machine in the *Host* parameter. This information is maintained by the **mountd** daemon on the *Host* parameter. This information is saved in the **/etc/rmtab** file in case the server crashes. The default value for the *Host* parameter is the value returned by the **hostname** command.

Note: If a client crashes, its entry will not be removed from the list until the client reboots and starts the **umount -a** command.

Note: The **showmount** command returns information maintained by the **mountd** daemon. Because NFS Version 4 does not use the **mount** daemon, **showmount** will not return information about version 4 mounts.

Flags

-a Prints all remote mounts in the format *HostName*:*Directory*, in which *HostName* is the name of the client and *Directory* is a directory pathname that has been remotely mounted.
-d Lists only directories that have been remotely mounted by clients.
-e Prints the list of exported directories.

Examples

1. To display a list of all remote directories mounted by a host, enter:

   ```
   /usr/bin/showmount  -a zeus
   ```

   In this example, the `showmount` command produces a list of all of the remote directories mounted by the clients on the host machine named `zeus`.
2. To display a list of only the directories mounted by a client on the host, enter:

   ```
   /usr/bin/showmount  -d athena
   ```

   In this example, the `showmount` command produces a list of all remote directories mounted by the client machines on the host named `athena`.
3. To print a list of all directories exported from a machine, enter:

   ```
   /usr/bin/showmount  -e zeus
   ```

   In this example, the `showmount` command produces a list of all remote directories exported by the host machine named `zeus`.

Files

/etc/rmtab Contains information about the current state of all exported directories.
/etc/xtab Lists currently exported directories.

Related Information

The **hostname** command, **umount** command.

The **mountd** daemon.

List of NFS commands.

Network file system, NFS troubleshooting in *Networks and communication management*.

shutacct Command

Purpose
Turns off processing accounting.

Syntax
/usr/sbin/acct/shutacct [*"Reason"*]

Description
The **shutacct** command turns off process accounting and calls the **acctwtmp** command to add a record stating the reason to the **/var/adm/wtmp** file. The **shutacct** command is invoked by the **shutdown** command.

Note: It is necessary to place quotation marks around the *Reason* value in the **/var/adm/wtmp** file.

Variables

Reason Specifies the reason for accounting system shutdown. This value is optional.

Security
Access Control: This command should grant execute (x) access only to members of the adm group.

Files

/usr/sbin/acct	The path to the accounting commands.
/var/adm/wtmp	The login and logout history file.

Related Information
The **turnacct** command.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in *Operating system and device management*.

Setting up an accounting subsystem in *Operating system and device management* explains the steps you must take to establish an accounting system.

shutdown Command

Purpose
Ends system operation.

Syntax
shutdown [**-d**] [**-F**] [**-h**] [**-i**] [**-k**] [**-l**] [**-m**] [**-p**] [**-r**] [**-t** *mmddHHMM* [*yy*]] [**-u**] [**-v**] [**+***Time* [*Message*]]

Description
The **shutdown** command halts the operating system. Only a user with root user authority can run this command. During the default shutdown, users are notified (by a **wall** command) of the impending system shutdown with a message. However, shutdown is not complete until the user receives a shutdown

completion message. Do not attempt to restart the system or turn off the system before the shutdown completion message is displayed; otherwise, file system damage can result.

> **Note:** The `halt completed` message is not displayed on the tty from which shutdown is invoked if it is connected to the system through a multiport adapter.

As shutdown time approaches, warning messages are displayed on the terminals of all users on the system.

After the specified number of seconds (60 by default), the system stops the accounting and error logging processes and writes an entry to the error log. The **shutdown** command then runs the **killall** command to end any remaining processes and runs the **sync** command to flush all memory resident disk blocks. Finally, it unmounts the file systems and calls the **halt** command.

> **Note:** Users who have files open on the node that is running the **shutdown** command, but who are not logged in to that node, are not notified about the shutdown.

If you request a complete halt to the operating system, the **shutdown** command stops all processes, unmounts all file systems, and calls the **halt** command.

The system administrator can place local customized shutdown procedures in a shell script named **/etc/rc.shutdown**. This script runs at the beginning of the shutdown if it exists. If the script runs but fails with a non-zero return code, the shutdown stops.

> **Attention:** If you are bringing the system down to maintenance mode, you must run the **shutdown** command from the **/** (root) directory to ensure that it can cleanly unmount the file systems.

> **Note:** By default, if issued on models having a power supply capable of software control, the **shutdown** command powers down the system.

Flags

-d Brings the system down from a distributed mode to a multiuser mode.

-F Does a fast shutdown, bypassing the messages to other users and bringing the system down as quickly as possible.

-h Halts the operating system completely; same as the **-v** flag.

-i Specifies interactive mode. Displays interactive messages to guide the user through the shutdown.

-k Allows the administrator to broadcast the **shutdown** warning messages *without* causing the system to shut down. When the **-k** flag is used, no other shutdown activity occurs except for sending messages. For example, no processes are killed, no activity is logged in **/etc/shutdown.log** if the **-l** flag is specified, and if an **/etc/rc.shutdown** script exists it does not run.

-l Creates/appends the **/etc/shutdown.log** file that contains information about the filesystems, daemons, user login, licensing services, network interfaces being brought down. The file may be used for diagnostic and debugging purposes in the event of shutdown failures.

> **Note:** Ensure that there is enough disk space for the **shutdown** command to log the entries while using this flag.

-m Brings the system down to maintenance (single user) mode.

-p Halts the system without a power down. This is used by uninterruptible power supply (UPS). This flag only applies to AIX 4.2 or later.

> **Note:** The **-p** flag will have no effect if used in combination with flags not requiring a permanent halt. Power will still be turned off if other operands request a delayed power-on and reboot

-r Restarts the system after being shutdown with the **reboot** command.

-t *mmddHHMM* [*yy*] Shuts down the system immediately and then restarts the system on the date specified by *mmddHHMM* [*yy*] where

mm Specifies the month.

dd Specifies the day.

HH Specifies the hour.

MM Specifies the minute.

yy Specifies the year.

The **shutdown -t** flag cannot be used with the **-v** or **-h** option.

> **Note:** This option is only supported on systems that have a power supply which automatically turns power off at shutdown and an alarm to allow reboot at a later time. Systems without this capability may hang or may reboot immediately after shutdown.

-u This flag is used by diagnostics to update the flash-memory and reboot.

-v Halts the operating system completely.

Parameters

+*Time* Specifies the time at which the **shutdown** command stops the system. An immediate shutdown is indicated by the word `now` displayed on the screen. A future time can be specified in one of two formats: +number or hour:minute. The first form brings the system down in the specified number of minutes and the second brings the system down at the time of day indicated (as a 24-hour clock). If the *Message* parameter is specified, the *Time* parameter must also be specified.

Message Specifies the message

Examples

1. To turn off the machine, enter:

   ```
   shutdown
   ```

   This shuts down the system, waiting 1 minute before stopping the user processes and the **init** process.
2. To give users more time to finish what they are doing and bring the system to maintenance mode, enter:

   ```
   shutdown  -m +2
   ```

   This brings the system down from multiuser mode to maintenance mode after waiting 2 minutes.

Files

/usr/sbin/shutdown Contains the **shutdown** command.

Related Information

The **errpt** command, **init** or **telinit** command, **kill** command, **killall** command, **halt** command, **reboot** command, and **sync** command.

The **sigaction** subroutine.

size Command

Purpose

Displays the section sizes of the Extended Common Object File Format (XCOFF) object files.

Syntax

size [**-d** | **-o** | **-x**] [**-f**] [**-V**] [**-X** {**32**|**64**|**32_64**}] *File* [*File ...*]

Description

The **size** command writes to standard output the number of bytes required by all sections, along with their sum for each XCOFF file. If the **-f** flag is specified, the section name follows the section size.

Flags

The output is in decimal notation unless you change the output with the following flags:

-d Writes in decimal notation.

-f Writes the section name in parenthesis following the section size.

-o Writes in octal notation.

-x Writes in hexadecimal notation.

-X *mode* Specifies the type of object file **size** should examine. The *mode* must be one of the following:

32 Processes only 32-bit object files

64 Processes only 64-bit object files

32_64 Processes both 32-bit and 64-bit object files

The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **size** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

-V Prints the version number of the **size** command.

Examples

1. To display the size of the **a.out** file in decimal, enter:

   ```
   size
   ```

   This displays the size in bytes of the executable **a.out** file. The size of each section of the object file is given, followed by the total:

   ```
   3720 + 1752 + 4152 = 9624
   ```

2. To display the size of an object file in octal, enter:

   ```
   size -o driver.o
   ```

   This displays the size of the **driver.o** object file in octal.

3. To display the size of several object files in hexadecimal, enter:

   ```
   size -x *.o
   ```

   This displays in hexadecimal the size of each file ending with **.o** in the current directory.

Related Information

The **ar** command, **as** command, **dump** command, **ld** command, **nm** command, **strip** command.

skulker Command

Purpose
Cleans up file systems by removing unwanted files.

Syntax
skulker

Description
> **Attention:** Because the **skulker** command is run by a root user, and its whole purpose is to remove files, it has the potential for unexpected results. Before installing a new **skulker** command, test any additions to its file removal criteria by running the additions manually using the **xargs -p** command. After you have verified that the new **skulker** command removes only the files you want removed, you can install it.

The **skulker** command is used for periodically purging obsolete or unneeded files from file systems. Candidate files include files in the **/tmp** directory, files older than a specified age, and the following file types: ***.bak**, **a.out**, **core**, proof, galley, **...***, **ed.hup**, and files that are more than one day old.

The **skulker** command is normally invoked daily, often as part of an accounting procedure run by the **cron** command during off-peak periods. Modify the **skulker** command to suit local needs following the patterns shown in the distributed version. Local users should be made aware of the criteria for automatic file removal.

The **find** command and the **xargs** command form a powerful combination for use in the **skulker** command. Most file selection criteria can be expressed conveniently with **find** expressions. The resulting file list can be segmented and inserted into **rm** commands using the **xargs** command to reduce the overhead that would result if each file were deleted with a separate command.

Related Information
The **cron** daemon, **find** command, **rm** command, **xargs** command.

slattach Command

Purpose
Attaches serial lines as network interfaces.

Syntax
/usr/sbin/slattach *TTYName* [*BaudRate DialString* [*DebugLevel*]]

Description
The **/usr/sbin/slattach** command assigns a TTY line to a network interface.

The **slattach** command is run by the **/etc/rc.net** file during system startup to automatically configure any Serial Line Internet Protocol (SLIP) network interfaces defined by the System Management Interface Tool (SMIT). SLIP interfaces can also be configured manually as shown in the examples section.

For a directly connected SLIP interface, broken connections are retried automatically without manual intervention. For a SLIP interface connected by modem, broken connections must be manually redialed. If a user supplies a dial string in the **slattach** command line, the user must re-enter the command and dial string to restore a broken connection.

To detach the interface, run the **ifconfig** *Interface* **down** command after terminating the **slattach** command. The *Interface* parameter is the name shown by the **netstat** command.

If configuring a slip interface from the command line, the **/usr/sbin/ifconfig** command must be invoked for the slip interface with the appropriate parameters and the slip tty line discipline must also be available in order for this command to succeed. To check if the slip tty line discipline is already loaded, run the command `strinfo -m | grep slip`. If no output is shown, the module has not yet been loaded. Load the module by issuing the command `strload -m /usr/lib/drivers/slip`.

Notes:

1. After the SLIP interface has been configured with **ifconfig**, any user who has permission on the TTY may issue the **slattach** command.
2. You must configure the tty devices used by the **slattach** command before establishing a connection. You may also need to make an entry for the tty device in the BNU **/usr/lib/uucp/Devices** file.
3. Sample shell script, **/usr/sbin/slipcall**, provides a simplified interface for invoking **slattach** and connecting to remote systems. **slipcall** is useful for connecting to dial-in SLIP networks that require a user to login before activating the SLIP tty line discipline. The basic configuration of **slipcall** will connect to other operating systems with **sliplogin** configurations and derive the local and remote internet addresses and network mask assigned by the called system. It then configures the local interface with the remote system's specified values.

Parameters

BaudRate	Sets the speed of the connection. The default speed is 9600.
DebugLevel	Sets the level of debug information desired. A number from 0 through 9 may be specified. A value of 0 specifies no debug information; a value of 9 specifies the most debug information. The default value is 0.
DialString	Specifies a string of expect/respond sequences using the Basic Networking Utility (BNU)/UNIX to UNIX Copy Program (UUCP) **chat** syntax.
TTYName	Specifies a TTY line. This string is in the form `ttyxx` or `/dev/ttyxx`.

Examples

1. To attach the SLIP network interface to the `tty1` port with a direct connection, issue the following command:

   ```
   slattach /dev/tty1
   ```

   This command attaches `tty1` to a network interface to be used by the SLIP.
2. To attach the SLIP network interface to `tty1` using a modem connection, issue the following command:

   ```
   slattach /dev/tty1 9600 '""AT OK \pATF1 OK \pATDT34335 CONNECT""'
   ```

Files

/etc/uucp/Devices	Lists definitions of devices used for remote connections.

Related Information

The **ifconfig** command, **netstat** command, **sliplogin** command.

TCP/IP network interfaces in *Networks and communication management*.

sleep Command

Purpose
Suspends execution for an interval.

Syntax
sleep *Seconds*

Description
The **sleep** command suspends execution of a process for at least the interval specified by the *Seconds* parameter. The amount of time specified in the *Seconds* parameter can range from 1 to **MAXINT** (2,147,483,647) seconds.

Exit Status
This command returns the following exit values:

0 The execution was successfully suspended for at least *Seconds* seconds, or a **SIGALRM** signal was received.
>0 An error occurred.

Examples
1. To run a command after a certain amount of time has passed, enter:

```
(
echo  "SYSTEM SHUTDOWN IN 10 MINUTES!" | wall
sleep 300; echo "SYSTEM SHUTDOWN IN 5 MINUTES!" | wall
sleep 240; echo "SYSTEM SHUTDOWN IN 1 MINUTE!" | wall
sleep 60; shutdown
)&
```

   This command sequence warns all users 10 minutes, 5 minutes, and 1 minute before the system is shut down.
2. To run a command at regular intervals, enter:

```
while true
do
 date
 sleep 60
done
```

   This shell procedure displays the date and time once a minute. To stop it, press the Interrupt key sequence.

Related Information
The **shutdown** command, **wall** command.

The **alarm** subroutine, **pause** subroutine, **sigaction** subroutine, **sleep** subroutine.

Shells in *Operating system and device management*.

slibclean Command

Purpose
Removes any currently unused modules in kernel and library memory.

Syntax

slibclean

Description

The **slibclean** command unloads all object files with load and use counts of 0. It can also be used to remove object files that are no longer used from both the shared library region and in the shared library and kernel text regions by removing object files that are no longer required.

Files

/usr/sbin/slibclean	Contains the **slibclean** command.

Related Information

The **unload** subroutine.

Using Kernel Processes in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

---

sliplogin Command

Purpose

Converts a standard-input terminal line into a Serial Line Internet Protocol (SLIP) link to a remote host.

Syntax

sliplogin [*LoginName*]

Description

The **sliplogin** command configures a standard-input terminal line into a Serial Line Internet Protocol (SLIP) link to a remote host; that is, the command attaches a serial line network interface.

Note: User requires root authority to attach a network interface.

The **sliplogin** command searches the **/etc/slip.hosts** file for a loginname entry that matches the value of the *LoginName* parameter. If a matching entry is found, **sliplogin** configures the line appropriately for SLIP (that is, for 8-bit transparent input/output) and converts it to SLIP line discipline. Then, **sliplogin** invokes the applicable login shell script which initializes the SLIP interface with the local and remote Internet Protocol (IP) addresses, netmask, and optional arguments associated with the loginname entry in the **/etc/slip.hosts** file.

The usual initialization script file is **/etc/slip.login**. However, in order to accommodate special initialization needs of a particular host, a script file named **/etc/slip.login.*userlogin*** (where *userlogin* corresponds to the loginname entry in the **/etc/slip.hosts** file) can be created. The **sliplogin** command uses the **/etc/slip.login.*userlogin*** script file when it exists, instead of the **/etc/slip.login** script file.

To deinitialize the SLIP interface, the **sliplogin** command uses either the **/etc/slip.logout** script file or the **/etc/slip.logout.*userlogin*** script file, if one of them exists, with preference given to the latter. The **/etc/slip.logout** script file is given the same arguments as the **/etc/slip.login** script file; the **/etc/slip.logout.*userlogin*** script file is given the same arguments as the **/etc/slip.login.*userlogin*** script file. In its default form, the **/etc/slip.logout** script file deletes all routes through the network interface for the specified SLIP unit. Additional processes to be done when the SLIP interface is disconnected can be added to either logout script file.

Notes:

1. The interface automatically deactivates when the remote connection terminates or when the **sliplogin** command dies.
2. Use the **slattach** command to access a remote system that has a SLIP link configured. Use the sample shell script file **/usr/sbin/slipcall** to invoke the **slattach** command with the proper parameters needed to call a remote system and configure the local interface with the appropriate values assigned by the remote system.
3. When using **sliplogin** as a user's login shell on a tty device, then this tty port used needs to be enabled for login. (This differs from the configuration when using **slattach** instead of **sliplogin** as a SLIP server process.

/etc/slip.hosts File

The **/etc/slip.hosts** file is the configuration file containing the names of preconfigured sliplogin users and the IP addresses to be assigned to the local and remote interface when the user logs in. **sliplogin** searches this file for matching *LoginName* entries. This file has the following format:

- Comments (lines starting with a #) and blank lines are ignored.
- Other lines must start with a *loginname* argument, and the fields should contain whatever is appropriate for the **slip.login** file that is executed for that name.
- Arguments are separated by white space and follow normal sh(1) quoting conventions. However, the *loginname* argument cannot be quoted. Usually lines have the following form:

  ```
  loginname local_address remote_address netmask opt_args
  ```

  where *local_address* and *remote_address* are the IP host names or addresses of the local and remote ends of the SLIP line, and *netmask* is the appropriate IP netmask. These arguments are passed directly to the **ifconfig** command. *Opt_args* are optional arguments used to configure the line.
- This implementation of **sliplogin** allows the **/etc/slip.hosts** file to contain multiple entries for a single SLIP user with differing addresses. This enables multiple SLIP interfaces to be activated by the **sliplogin** command for the same user name. When user entries are retrieved from the **/etc/slip.hosts** file, only entry addresses meeting the following criteria are selected.

  The entry is ignored if a `slip.hosts` entry specifies a local address which is already in use on another non-SLIP interface on the local system.

  The entry is ignored if the remote address specified in an **/etc/slip.hosts** entry is already in use on any other interface.

/etc/slip.login File

The **/etc/slip.login** or **/etc/slip.login.*userlogin*** file is the setup script invoked by the **sliplogin** command to initialize the user's network interface. The **/etc/slip.login.*userlogin*** file is invoked if it exists, where the value of the *LoginName* parameter of the **sliplogin** command corresponds to a loginname entry in the **/etc/slip.hosts** file. If this file cannot be accessed, the **/etc/slip.login** file is invoked instead. The login script file contains the following parameters:

slipunit	Specifies the unit number of SLIP interface assigned to this line. For example, 0 for sl0 (sl0 is s, lowercase L, zero.)
speed	Specifies the speed of the line.
args	Specifies the arguments from the **/etc/slip.hosts** file entries, in order, starting with *loginname*.

/etc/slip.logout File

The **/etc/slip.logout** or **/etc/slip.logout.*userlogin*** file is the setup script invoked by **sliplogin** to deinitialize the user's network interface. The **/etc/slip.logout.*userlogin*** file is invoked if it exists, where the value of the *LoginName* parameter of **sliplogin** corresponds to a loginname entry in the **/etc/slip.hosts** file. If this file cannot be accessed, the **/etc/slip.logout** file is invoked instead.

Flags

</dev/ttyx Redirects the command to the **ttyx** device if the user is already logged into a **tty** device and wants to configure their terminal as a SLIP line.

Parameters

LoginName Specifies the desired login name. The default is the current login name.

Examples

The normal use of the **sliplogin** command is to create an **/etc/passwd** entry for each legal, remote SLIP site with **sliplogin** as the shell for the entry. For example,

```
foo:!:2010:1:slip line to foo:/tmp:/usr/sbin/sliplogin
```

An entry must then be added to the **/etc/slip.hosts** file. The entry should resemble the following example:

```
foo 1.1.1.1 1.1.1.2 0xffffff00 normal
```

where *loginname* = `foo`, *local_address* = `1.1.1.1`, *remote_address* = `1.1.1.2`, *netmask* = `0xffffff00`, and `opt_args = normal`. (The optional argument `normal` indicates which SLIP mode to activate. For AIX 4.1, only normal mode is supported.)

Diagnostics

The **sliplogin** command logs various information to the system log daemon (**syslogd**). The messages are listed here, grouped by severity levels.

Error Severity

Message	Description
ioctl (TCGETS): *reason*	The **ioctl** subroutine failed to get the line parameters for the reason indicated.
ioctl (TCSETS): *reason*	The **ioctl** subroutine failed to set the line parameters for the reason indicated.
ioctl (TIOCGETD): *reason*	The **ioctl** subroutine failed to get the current **tty** discipline for the reason indicated.
/etc/slip.hosts: *reason*	The **/etc/slip.hosts** file could not be opened for the reason indicated.
Check of flags for interface *xxx* failed. Errno is reason.	An attempt to check the status of the indicated interface to avert possible addressing conflicts failed for the reason indicated in the **errno** global variable.
Access denied for user - no /etc/slip.login[.***userlogin***] file.	No **/etc/slip.login** or **/etc/slip.login.*userlogin*** script file could be found.
Access denied for user - no /etc/slip.hosts entries available.	No loginname entry in the **/etc/slip.hosts** file matched the *LoginName* value specified in the command.
Access denied - getlogin returned 0.	The user issuing the **sliplogin** command does not have a password entry in the **/etc/passwd** file.
Logout script failed: exit status *xxx* from /etc/slip.logout[.***userlogin***]	An attempt to run the **/etc/slip.logout** or **/etc/slip.logout.*userlogin*** script file failed with the indicated exit status.

Error Severity

Message	Description
No SLIP interface for ttyx. Errno is reason.	No SLIP interface could be located for the **ttyx** device for the reason indicated in the **errno** global variable. Try either running the **ifconfig slx up** command or using SMIT to add a network interface for the **tty** device.
Open /dev/null: *reason*	An attempt to open the **/dev/null** device failed for the reason indicated.
/etc/slip.logout file not found	The **/etc/slip.logout** file could not be located.
sliplogin: cannot add SLIP discipline to ttyx	No SLIP interface exists for the **ttyx** device. Try either running the **ifconfig slx up** command or using SMIT to add a network interface for the **tty** device.
SLIP discipline removal from tty failed. Errno is reason.	An attempt to remove the SLIP discipline from the **tty** device failed for the reason indicated in the **errno** global variable.
tcgetattr: *reason*	An attempt to read the current attributes of the **tty** device failed for the reason indicated.
userlogin login failed: exit status *xxx* from /etc/slip.login[.*userlogin*]	A system call to execute the **/etc/slip.login** or **/etc/slip.login.*userlogin*** script file failed with the indicated exit status.

Information Severity

Message	Description
Attaching SLIP unit *xxx* for *userlogin* on ttyx.	The **sliplogin** command found a loginname entry in the **/etc/slip.hosts** file that matched the *LoginName* value specified in the command, invoked the applicable **/etc/slip.login** or **/etc/slip.login.*userlogin*** file, and is now attaching the indicated network interface.
Closed *userlogin* SLIP unit *xxx* (signal)	The indicated SLIP unit for the indicated *userlogin* was closed because the **sliplogin** command terminated due to a signal.

Notice Severity

Message	Description
Attaching SLIP unit *xxx* for *userlogin*.	The indicated SLIP unit has been successfully attached for the indicated *userlogin*.

Files

/etc/slip.hosts	The configuration file that contains the names of preconfigured sliplogin users and the IP addresses to be assigned to the local and remote interface when the user logs in.
/etc/slip.login or **/etc/slip.login.*userlogin***	The setup script invoked by the **sliplogin** command to initialize the user's network interface.
/etc/slip.logout or **/etc/slip.logout.*userlogin***	The setup script invoked by the **sliplogin** command to deinitialize the user's network interface.

Related Information

The **slattach** command.

slocal Command

Purpose

Processes incoming mail.

Syntax

slocal [**-verbose** | **-noverbose**] [**-debug**]

Description

The **slocal** command performs a set of actions each time a message is sent to the user. The **slocal** command is not started by the user. The **slocal** command is called by the **sendmail** command.

The **sendmail** command starts the **slocal** command upon encountering the following line in the **$HOME/.forward** files:

```
/usr/lib/mh/slocal
```

For each incoming message, the **slocal** command performs the actions specified in the **.maildelivery** file. If the **slocal** command cannot find the **$HOME/.maildelivery** file, the **slocal** command uses the **/etc/mh/maildelivery** default file. If the delivery request fails, the **slocal** command delivers the message to the **/usr/mail/$USER** file.

Flags

-debug	Provides information for debugging.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-noverbose	Does not display information as the system executes commands in the **.maildelivery** file. This flag is the default.
-verbose	Displays information as the system executes commands in the **.maildelivery** file.

Files

/usr/lib/mh/mtstailor	Contains MH command definitions.
/etc/mh/.maildelivery	Contains the default MH instructions for local mail delivery.
$HOME/.maildelivery	Provides the user with MH instructions for local mail delivery.
$HOME/.forward	Contains either the line that starts the **slocal** command or a path to forward mail.
/etc/mh/mh_profile	Contains parameters that customize the MH package.

Related Information

The **rcvdist** command, **rcvpack** command, **rcvstore** command, **rcvtty** command, **sendmail** command.

Mail applications in *Networks and communication management*.

smcaprop Command

Purpose

Provides read-only information on the Certificate Authority.

Syntax

smcaprop

Description

The **smcaprop** command can be run on a machine that has been defined as internal Certificate Authority (CA). The command prompts for the CA private key ring password, and then provides read-only information on the CA (CA Name, Most recent certificate issued, CA certificate expiration date, etc.). Detailed information on all operations executed by the CA (key ring generation, certificate signing, etc.) can be found in the CA log file **/usr/websm/security/SMCa.log**.

You can use Web-based System Manager (wsm) command to access the graphical interface.

Examples

```
smcaprop
```

Files

/usr/websm/security/SMCa.log	Lists detailed information on all operations executed by the CA.
/usr/websm/security/SM.caprivkr	Certificate private key ring file.

Related Information

The **smdefca**, **smexpcacert**, **smimpcacert**, **smlistcerts**, **smsigncert**, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smdefca Command

Purpose

Defines an internal certificate authority.

Syntax

smdefca *ca_name* **-o** *organization* **-c** *country_code* **-d** *pub_dir* [**-e** *mm/dd/yyyy*]

Description

The **smdefca** command is used to define an internal CA (Certificate Authority) for Web-based System Manager servers and clients on the current machine. When you define a Web-based System Manager-CA, the following files are generated:

/usr/websm/security/SM.caprivkr
: This is the CA private key ring that includes the CA private key and the CA certificate. This is the most sensitive file from the aspect of Web-based System Manager security. It is created **root** protected and password encrypted.

SMpubkr.class (created on the specified *pub_dir***)**
: The public key ring file. This file has to be distributed to each Web-based System Manager client (for application mode) and server (for applet mode) and should be placed in **/usr/websm/codebase**.

If a CA is already defined on the current machine, the **smundefca** command must be used first to unconfigure it.

Use the **/usr/websm/bin/wsm** command to access the graphical interface. The fast path is **wsm system**.

Flags

ca_name
A name that uniquely defines your Web-based System Manager-CA. The machine full TCP/IP name with some additional serial number might be a good choice. If you ever redefine a CA, it is recommended that you use a different name in order to identify which CA, by name, is used by each server and client.

> **Note:** Do not set the CA name to be exactly the machine's full TCP/IP name (this will break the SMGate utility, in case you want to use it in managing this machine from a remote browser).

-o *organization*
Organization name (required for the CA certificate).

-c *country_code*
Two-letter ISO country code (required for the CA certificate).

-d *pub_dir*
The output directory for the public key ring file **SMpubkr.class**.

-e *mm/dd/yyyy*
Expiration date for the CA certificate. The default expiration date is four years from the date of issuing the command.

Examples

```
smdefca IBMCA1 -o IBM -c US -d /usr/websm/security/tmp -e 12/31/1999
```

Files

/usr/websm/security/SMpubkr.class
CA public key ring file.

/usr/websm/security/SMCa.log
Lists detailed information on all operations executed by the CA.

/usr/websm/security/SMCa.sn
Certificate number file.

/usr/websm/security/SM.caprivkr
Certificate private key ring file.

Related Information

The **smcaprop**, **smexpcacert**, **smimpcacert**, **smlistcerts**, **smsigncert**, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smdemon.cleanu Command

Purpose

Cleans up the **sendmail** queue for periodic housekeeping.

Syntax

/usr/lib/smdemon.cleanu

Description

The **smdemon.cleanu** command is a shell procedure that cleans up the **sendmail** command queue and maintains the **/var/spool/mqueue/log** file.

To enable the **smdemon.cleanu** command, you must remove the comment statement by deleting the # character from the beginning of the **smdemon.cleanu** line in the **/var/spool/cron/crontabs/root** file. If the **/var/spool/mqueue** directory does not exist, do not change the **/var/spool/cron/crontabs/root** file.

Be careful that the average size of a log file for each **smdemon.cleanu** session multiplied by the number of log files does not use more space than you need. You can arrange the number of log files to suit your needs.

Note: The **smdemon.cleanu** command is not usually entered on the command line. The command is executed by the **cron** daemon.

Examples

To run the **smdemon.cleanu** procedure automatically, edit the **/var/spool/cron/crontabs/root** file and delete the # (comment character) from the beginning of the **smdemon.cleanu** line as follows:

```
# ulimit 5000; /usr/lib/smdemon.cleanu > /dev/null
```

Files

/var/spool/cron/crontabs/root	Schedules when the **smdemon.cleanu** command will run.
/var/spool/mqueue	Contains the **log** file and temporary files associated with the message in the mail queue.

Related Information

The **cron** daemon.

The **sendmail** command.

Mail logging and Log management in *Networks and communication management*.

---

smexpcacert Command

Purpose

Exports the certificate authority (CA) certificate.

Syntax

smexpcacert *cert_file*

Description

The **smexpcacert** command can be run on a machine that has been defined as internal certificate authority (CA). The command prompts for the CA private key ring password, and then writes the CA certificate of the internal CA to the file *cert_file*. The full path name of the output file for the CA certificate is specified with *cert_file*.

You can use Web-based System Manager command to access the graphical interface.

Examples

```
smexpcacert /tmp/CA1.cert
```

Files

/usr/websm/security/SMCa.log	Lists detailed information on all operations executed by the CA.

Related Information

The **smcaprop**, **smdefca**, **smimpcacert**, **smlistcerts**, **smsigncert**, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smgenkeycr Command

Purpose

Generates server private keys and certificate requests.

Syntax

smgenkeycr { *server_name* | **-f** *list_file*} **-o** *organization* **-c** *country_code* **-d** *out_dir* [**-k 512**]

Description

The **smgenkeycr** command generates a private key and a certificate request for the Web-based System Manager servers. The private keys and certificate requests are generated either for each server in the input server list file or for the server whose name is specified.

You can use Web-based System Manager command to access the graphical interface.

Flags

server_name	The full TCP/IP name of the server. If the name specified is **S**, a private key ring file **S.privkr** will be generated in the output directory.
-f *list_file*	The full path name of a file with the list of server machine names (one line with full TCP/IP name per server). For each server **S** in the list, a private key ring file **S.privkr** will be generated in the output directory.
-o *organization*	Organization name (required for the server certificate).
-c *country_code*	Two-letter ISO country code (required for the server certificate).
-d *out_dir*	The output directory for the server private key ring files.
-k 512	This option does not exist in the exportable version. The server private key length will be 512. The default in the US version is 1024, in the exportable - 512.

Examples

```
smgenprivkr S101.IBM.COM -o IBM -c US -d /usr/websm/security/tmp
smgenprivkr -f /usr/websm/security/tmp/server.list -o IBM -c US -d /usr/websm/security/tmp
```

Related Information

The **smgenprivkr**, **smimpservercert**, **sminstkey**, **smlistcerts**, and the **smserverprop** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smgenprivkr Command

Purpose

Generates server private key ring files.

Syntax

smgenprivkr { *server_name*| **-f** *list_file*} **-o** *organization* **-c** *country_code* **-d** *out_dir* [**-k 512**] [**-e** *mm/dd/yyyy*]

Description

The **smgenprivkr** command can be run on a machine that has been defined as internal certificate authority (CA). The **smgenprivkr** command generates 'ready to go' private key ring files for the Web-based System Manager servers. The private key ring files are generated either for each server in the input server list file, or for the one server whose name is specified.

You can use Web-based System Manager command to access the graphical interface.

Flags

server_name	The full TCP/IP name of the server. If the name specified is **S**, a private key ring file **S.privkr** will be generated in the output directory.
-f *list_file*	The full path name of a file with the list of server machine names (one line with full TCP/IP name per server). For each server **S** in the list, a private key ring file **S.privkr** will be generated in the output directory.
-o *organization*	Organization name (required for the server certificate).
-c *country_code*	Two-letter ISO country code (required for the server certificate).
-d *out_dir*	The output directory for the servers private key ring files.
-k 512	This option does not exist in the exportable version. The server's private key length will be 512. The default in the US version is 1024, in the exportable 512.
-e *mm/dd/yyyy*	Expiration date for the server certificates. The default expiration date is two years from the date of issuing the command.

Examples

```
smgenprivkr S101.IBM.COM -o IBM -c US -d /usr/websm/security/tmp -e 12/31/1999

smgenprivkr -f /usr/websm/security/tmp/server.list -o IBM -c US -d /usr/websm/security/tmp
```

Files

/usr/websm/security/SMCa.log	Lists detailed information on all operations executed by the CA.

Related Information

The **smgenkeycr**, **smimpservercert**, **sminstkey**, **smlistcerts**, and the **smsigncert** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smimpcacert Command

Purpose

Imports the certificate authority's certificate.

Syntax

smimpcacert *dir cert_file*

Description

The **smimpcacert** command imports the certificate authority (CA) certificate *cert_file* into the public key ring file **SMpubkr.class** that resides on the specified directory *dir*. If there is no **SMpubkr.class** file in *dir*, a new **SMpubkr.class** containing only the certificate of *cert_file* is created there.

You can use Web-based System Manager command to access the graphical interface.

Parameters

dir	The directory of **SMpubkr.class**.
cert_file	The full path name of the CA certificate file.

Examples

```
smimpcacert /usr/websm/security/tmp CA1.cert
```

Related Information

The **smcaprop** command, **smdefca** command, **smexpcacert** command, **smlistcerts** command, **smsigncert** command, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smimpservercert Command

Purpose

Imports the Server Certificate.

Syntax

smimpservercert *dir* { **-a** | **-s** *server_name*}

Description

The **smimpservercert** command imports a server certificate (***.cert** file) to the server private key file (***.privk**), generating a private key ring file (***.privkr**). You can import the certificate of one server, or all certificates with matching private key files in the specified directory *dir*.

Use the **/usr/websm/bin/wsm** command to access the graphical interface. The fast path is **wsm system**.

Flags

dir	The directory where the certificate requests (***.certreq** files) and the private keys (***.privk** files) reside, and to which the private key rings (***.privkr** files) will be written.
-a	All certificates (***.certreq** files) with matching private key files (***.privk**) in the specified *dir* directory will be processed. Each certificate **S.cert** will be imported into the private key file **S.privk**, generating the private key ring file **S.privkr** in the specified *dir* directory.
-s *server_name*	The full TCP/IP name of the server whose certificate *server_name***.cert** will be imported into its private key file *server_name***.privk**, generating the private key ring file *server_name***.privkr** in the specified *dir* directory.

Examples

```
smimpservercert /usr/websm/security/tmp S101.IBM.COM

smimpservercert /usr/websm/security/tmp -a
```

Related Information

The **smgenkeycr**, **smgenprivkr**, **sminstkey**, **smlistcerts**, and the **smserverprop** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

sminstkey Command

Purpose

Installs private key ring.

Syntax

sminstkey [*inpdir* | **-d** | *tarfile*]

Description

The **sminstkey** command expects the private key ring file of the current machine (**S.privkr** where **S** is the full TCP/IP machine name) in a directory, or on a diskette archive created by TAR, or in a TAR file, and installs it as **/usr/websm/security/SM.privkr**.

> **Note:** In case of TAR file or diskette TAR, the private key ring should appear there without a path.

If the source private key ring file is password-encrypted, the command prompts for the password.

You can use Web-based System Manager (wsm) command to access the graphical interface.

Flags

inpdir	The source **S.privkr** is in the directory *inpdir*.
-d	The source **S.privkr** is in a diskette archive created by TAR.
tarfile	The source **S.privkr** is in the TAR file *tarfile*.

Examples

```
sminstkey /usr/websm/security/tmp

sminstkey -d

sminstkey /afs/security/privkrs.tar
```

Files

/usr/websm/security/SM.privkr	Server private key ring file.

Related Information

The **smgenkeycr**, **smgenprivkr**, **smimpservercert**, **smlistcerts**, and the **smserverprop** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

smit Command

Purpose

Performs system management.

Syntax

smit [**-C** | **-M**] [**-D**] [**-f**] [**-h**] [**-l** *File*] [**-o** *PathName*] [**-p** *Entity/ValueString*] [**-r** *RunMode*] [**-s** *File*] [**-t**] [**-v**] [[**-m** | **-n** | **-d**] *FastPath*] [**-X**] [**-x**]

Description

The **smit** command invokes the System Management Interface Tool (SMIT). SMIT is an interactive interface application designed to simplify system management tasks. The **smit** command displays a hierarchy of menus that can lead to interactive dialogues. SMIT builds and runs commands as directed by the user. Because SMIT runs commands, you need the authority to execute the commands that SMIT runs.

SMIT creates two files, the **smit.script** file and the **smit.log** file. Invoking the **smit** command with the **-s** *PathName* flag saves the **smit.script** file in the file specified by the *PathName* parameter. If the **-s** flag is not specified, the script information is saved in the **$HOME/smit.script** file. Invoking the **smit** command with the **-l** *PathName* flag saves the **smit.log** file in the file specified by the *PathName* parameter. If the **-l** flag is not specified, the log information is recorded in the **$HOME/smit.log** file. You must have write permission for the directory in which you have requested the **smit** file to be written or the **smit.script** file and **smit.log** file are not created. SMIT does not overwrite the **smit.log** file or the **smit.script** file. The files are appended when possible.

The **smit.script** file automatically records the commands with the command flags and parameters used. The **smit.script** file can be used as an executable shell script to duplicate system configuration. SMIT creates the **smit.log** file, which contains additional detailed information that can be used by programmers in extending the SMIT system. The **smit.log** file is affected by the **-D**, **-l**, **-t**, and **-v** flags.

The **smit** command takes you to the top level of the menu hierarchy if you do not use the *FastPath* parameter. To enter the menu at lower levels, use the *FastPath* parameter. All commands run by SMIT can be used as *FastPaths*. The *FastPath* parameter will assist you as you become familiar with the commands. For example, you can enter: `smit chuser` to go directly to the dialog from which you can change user characteristics. To learn more about *FastPaths* see, Setting up and maintaining roles using SMIT in *Security*.

Note: User access to SMIT panels may be controlled with the **smitacl.user** or **smitacl.group** commands.

SMIT requires access to the following files:

sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are corrupt, or exist on an NFS server and that server goes down, SMIT may fail to respond.

Flags

-C Starts SMIT using an ASCII (also called Curses) interface.

-D Sets the debug mode; sets **-t** and **-v** flags.

-d *FastPath* Identifies that the *FastPath* is the name of a dialogue.

-f Allows standard in and standard out from SMIT to be redirected.

-h Displays the command usage message.

-l *File* Redirects the **smit.log** file to the specified *File*.

-M Starts SMIT using a windows (also called Motif) interface.

-m *FastPath* Identifies that the *FastPath* is the name of a menu.

-n *FastPath* Identifies that the *FastPath* is the name of a selector.

-o *PathName* Specifies a directory *PathName* of an alternate repository for SMIT objects. The default directory is **/etc/objrepos**.

-p *Entity/ValueString* This flag only applies to smit Windows® version. Allows nameselects and dialogs to be filled in from the command line. Also allows you to operate on multiple entities simultaneously. You can set the environment variables ENTITY_SEP and VALUE_SEP to override the default comma and semicolon separators.

You can enter *Entity/ValueString* in any of the following formats:

"Entity1:*Val1*,*Val2*... ; *Entity2*:*Val1*,*Val2*... ; ..."

or

"Val1,*Val2*... ; *Val1*,*Val2*... ; ..."

-r *RunMode* This flag only applies to smit Windows version. Specifies the mode to run msmit in.

You can enter the following values for *RunMode*:

1 Exit msmit when **done** is clicked in the output window.

2 Exit msmit when **ok** is clicked in a dialog. Print the dialog options upon exit. Do not run the command.

3 Run msmit silently, print the dialog options. Do not run the command.

4 Exit msmit when **ok** is clicked in the dialog. Print the commands upon exit. Do not run the command.

-s *File* Redirects the **smit.script** file to the specified *File*.

-t Records detailed trace information in the **smit.log** file.

-v Records the command strings for intermediate and target task commands run by SMIT, and also records their output in the **smit.log** file.

-x Does not run any **command_to_execute**, but still logs them for later execution.

-X Does not run any **command_to_discover**, **command_to_list**, **command_to classify** or **command_to_execute**.

Examples

1. To display the main menu in the overall system management hierarchy, enter:
   ```
   smit
   ```
2. To change the characteristics of a user, enter:
   ```
   smit chuser
   ```

   The **chuser** command is an example of a *FastPath* parameter. The **smit** command and the *FastPath* parameter **chuser** takes you directly to the dialog, `Change User Attributes`, which guides you through changing the characteristics of a user.

3. To make the **smit.script** file executable for duplicate configuration, enter:

   ```
   chmod +x smit.script
   ```

   Then, to duplicate your configuration, enter:

   ```
   smit.script
   ```

   The **smit.script** file can be edited to create slight variations in the configuration commands, or to use only subsets of the commands. The **smit.script** file should be renamed or copied to prevent SMIT from modifying it.

   > **Note:** SMIT runs commands under the Korn shell (**/usr/bin/ksh**). Some command strings in the **smit.script** file may require this environment to run correctly.

Files

/usr/bin/smit	Contains the **smit** command.
/etc/objrepos	Specifies the default directory for the SMIT database.
smit.log	Specifies detailed information of your session, with time stamps.
smit.script	Specifies only the target task commands run by SMIT, with time stamps.

Related Information

The **chmod**, **chsec**, and **lssec**, commands.

The **smitacl.group** file and the **smitacl.user** file.

System Management Interface Tool (SMIT) in *Operating system and device management*.

System Management Interface Tool (SMIT) Overview for Programming in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

smitty Command

Purpose

Provides a Curses-based text interface to perform system management.

Syntax

smitty [**-C**] [**-D**] [**-f**] [**-h**] [**-l** *File*] [**-o** *PathName*] [**-s** *File*] [**-t**] [**-v**] [[**-m** | **-n** | **-d**] *FastPath*] [**-X**] [**-x**]

Description

The **smitty** command invokes the System Management Interface Tool (SMIT). SMIT is an interactive interface application designed to simplify system management tasks. The **smitty** command displays a hierarchy of menus that can lead to interactive dialogues. SMIT builds and runs commands as directed by the user. Because SMIT runs commands, you need the authority to execute the commands that SMIT runs.

Note: The **smitty** command is identical to **smit -C**.

SMIT creates two files, the **smit.script** file and the **smit.log** file. Invoking the **smitty** command with the **-s** *PathName* flag saves the **smit.script** file in the file specified by the *PathName* parameter. If the **-s** flag is not specified, the script information is saved in the **$HOME/smit.script** file. Invoking the **smitty** command with the **-l** *PathName* flag saves the **smit.log** file in the file specified by the *PathName* parameter. If the **-l**

flag is not specified, the log information is recorded in the **$HOME/smit.log** file. You must have write permission for the directory in which you have requested the **smit** files to be written or the **smit.script** file and **smit.log** file are not created. SMIT does not overwrite the **smit.log** file or the **smit.script** file. The files are appended when possible.

The **smit.script** file automatically records the commands with the command flags and parameters used. The **smit.script** file can be used as an executable shell script to duplicate system configuration. SMIT creates the **smit.log** file, which contains additional detailed information that can be used by programmers in extending the SMIT system. The **smit.log** file is affected by the **-D**, **-l**, **-t**, and **-v** flags.

The **smitty** command takes you to the top level of the menu hierarchy if you do not use the *FastPath* parameter. To enter the menu at lower levels, use the *FastPath* parameter. All commands run by SMIT can be used as *FastPaths*. The *FastPath* parameter will assist you as you become familiar with the commands. For example, you can enter: `smitty chuser` to go directly to the dialog from which you can change user characteristics. To learn more about *FastPaths* see, Setting up and maintaining roles using SMIT in *Security*.

SMIT requires access to the following files:

sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are corrupt, or exist on an NFS server and that server goes down, SMIT may fail to respond.

Flags

-C	Starts SMIT using a Curses-based text interface. This is the default for the **smitty** command.
-D	Sets the debug mode; sets **-t** and **-v** flags.
-d *FastPath*	Identifies that the *FastPath* is the name of a dialogue.
-f	Allows standard in and standard out from SMIT to be redirected.
-h	Displays the command usage message.
-l *File*	Redirects the **smit.log** file to the specified *File*.
-m *FastPath*	Identifies that the *FastPath* is the name of a menu.
-n *FastPath*	Identifies that the *FastPath* is the name of a selector.
-o *PathName*	Specifies a directory *PathName* of an alternate repository for SMIT objects. The default directory is **/etc/objrepos**.
-s *File*	Redirects the **smit.script** file to the specified *File*.
-t	Records detailed trace information in the **smit.log** file.
-v	Records the command strings for intermediate and target task commands run by SMIT, and also records their output in the **smit.log** file.
-x	Does not run any **command_to_execute**, but still logs them for later execution.
-X	Does not run any **command_to_discover**, **command_to_list**, **command_to classify** or **command_to_execute**.

Examples

1. To display the main menu in the overall system management hierarchy, enter:

   ```
   smitty
   ```

2. To change the characteristics of a user, enter:

   ```
   smitty chuser
   ```

   The **chuser** command is an example of a *FastPath* parameter. The **smitty** command and the *FastPath* parameter **chuser** takes you directly to the dialog, `Change User Attributes`, which guides you through changing the characteristics of a user.

3. To make the **smit.script** file executable for duplicate configuration, enter:

   ```
   chmod +x smit.script
   ```

   Then, to duplicate your configuration, enter:

   ```
   smit.script
   ```

   The **smit.script** file can be edited to create slight variations in the configuration commands, or to use only subsets of the commands. The **smit.script** file should be renamed or copied to prevent SMIT from modifying it.

   > **Note:** SMIT runs commands under the Korn shell (**/usr/bin/ksh**). Some command strings in the **smit.script** file may require this environment to run correctly.

Files

/usr/bin/smitty	Contains the **smitty** command.
/etc/objrepos	Specifies the default directory for the SMIT database.
smit.log	Specifies detailed information of your session, with time stamps.
smit.script	Specifies only the target task commands run by SMIT, with time stamps.

Related Information

The **chmod** command.

System Management Interface Tool (SMIT) in *Operating system and device management*.

System Management Interface Tool (SMIT) Overview for Programming in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

smlistcerts Command

Purpose

Lists CA certificates.

Syntax

smlistcerts *dir*

Description

The **smlistcerts** command lists the CA certificates contained in the public key ring file **SMpubkr.class** that resides on the specified directory *dir*. The directory of **SMpubkr.class** is specified by *dir*.

You can use Web-based System Manager (wsm) command to access the graphical interface.

Examples

```
smlistcerts /usr/websm/codebase
```

Related Information

The **smcaprop**, **smdefca**, **smexpcacert**, **smimpcacert**, **smsigncert**, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

smrsh Command

Purpose

Restricted shell for sendmail.

Syntax

smrsh -c *command*

Description

The **smrsh** command is intended as a replacement for the **sh** command in the prog mailer in **sendmail** configuration files. The **smrsh** command limits the programs that can be run using the **sendmail** command syntax. This improves overall system security. **smrsh** limits the set of programs that a programmer can execute, even if **sendmail** runs a program without going through an alias or forward file.

The **smrsh** command requires that programs be in the **/var/adm/sm.bin** directory. This allows system administrators to choose which programs can be run by the **smrsh** command. The **smrsh** command also rejects any commands with the following characters on the command line to prevent end-run attacks: ,, <, >, |, ;, &, $, \r (<RETURN>), or \n (<NEWLINE>) on the command line to prevent end run attacks.

- ,
- <
- >
- |
- ;
- &
- $
- \r (<RETURN>)
- or \n (<NEWLINE>)

Initial pathnames on programs are stripped, so forwarding to **/usr/ucb/vacation**, **/usr/bin/vacation**, **/home/server/mydir/bin/vacation**, and **vacation** all actually forward to **/var/adm/sm.bin/vacation**. System administrators should be conservative about populating **/var/adm/sm.bin**. Reasonable additions are utilities such as **vacation(1)** and **procmail**. Never include any shell or shell-like programs (for example, **perl**) in the **sm.bin** directory. This does not allow the execution of arbitrary programs, but does not restrict the use of shell or perl scripts in the **sm.bin** directory (using the **#!** syntax).

Flags

-c *command*
: Runs the program specified by *command*.

Location

/usr/sbin/smrsh
Default location of smrsh command.

Files

/var/adm/sm.bin
Directory for restricted programs.

Related Information

The **bellmail** command, **kill** command, **mail, Mail** command, **mailq** command, **mailstats** command, **newaliases** command, **refresh** command, **sendmail** command, and**uux** command.

The **srcmstr** daemon.

Basic Networking Utilities, Mail management, and Transmission Control Protocol/Internet Protocol in *Networks and communication management*

Mail applications in *Networks and communication management*.

---

smserverprop Command

Purpose

Lists server properties.

Syntax

smserverprop

Description

The **smserverprop** command provides read-only information on the local Web-based System Manager server (Name, key length, certificate expiration date, Certificate Authority name etc.).

You can use Web-based System Manager (wsm) command to access the graphical interface.

Examples

```
smserverprop
```

Files

/usr/websm/security/SM.privkr	Server private key ring file.

Related Information

The **smgenkeycr**, **smgenprivkr**, **smimpservercert**, **sminstkey**, and the **smlistcerts** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

smsigncert Command

Purpose

Processes certificate requests and generates certificates.

Syntax

smsigncert *dir* { **-a** | **-s** *server_name*} [**-e** *mm/dd/yyyy*]

Description

The **smsigncert** command can be run on a machine that has been defined as internal certificate authority (CA). The command uses the CA private key to process certificate requests (***.certreq** files) and generate certificates (***.cert** files). You can process the request of one server, or all server requests in the specified directory *dir*.

You can use Web-based System Manager (wsm) command to access the graphical interface.

Flags

dir	The directory where the certificate requests (***.certreq** files) reside, and to which the certificates (***.cert** files) will be written.
-a	All certificate requests (***.certreq** files) in the specified *dir* directory will be processed. For each certificate request (**S.certreq**), a certificate **S.cert** will be generated in the specified *dir* directory.
-s *server_name*	The full TCP/IP name of the server whose certificate request (*server_name***.certreq** in the specified *dir* directory) will be processed.
-e *mm/dd/yyyy*	Expiration date for the server certificates. The default expiration date is two years from the date of issuing the command.

Examples

```
smsigncert /usr/websm/security/tmp S101.IBM.COM -e 12/31/1999

smsigncert /usr/websm/security/tmp -a
```

Files

/usr/websm/security/SMCa.log	Lists detailed information on all operations executed by the CA.

Related Information

The **smcaprop**, **smdefca**, **smexpcacert**, **smlistcerts**, and the **smundefca** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

smtctl Command

Purpose

The **smtctl** command controls the enabling and disabling of processor simultaneous multi-threading mode.

Syntax

smtctl [**-m off** | **on** [**-w boot** | **now**]]

Description

This command is provided for privileged users and applications to control utilization of processors with simultaneous multi-threading support. The simultaneous multi-threading mode allows processors to have thread level parallelism at the instruction level. This mode can be enabled or disabled for all processors either immediately or on subsequent boots of the system. This command controls the simultaneous multi-threading options.

Flags

-m off	This option will set simultaneous multi-threading mode to `disabled`.
-m on	This option will set simultaneous multi-threading mode to `enabled`.
-w boot	This option makes the simultaneous multi-threading mode change effective on next and subsequent reboots if you run the **bosboot** command before the next system reboot.
-w now	This option makes the simultaneous multi-threading mode change immediately but will not persist across reboot. If neither the **-w boot** or the **-w now** options are specified, then the mode change is made immediately. It will persist across subsequent reboots if you run the **bosboot** command before the next system reboot.

If no options are specified then the following simultaneous multi-threading settings will be reported:

SMT Capability	Indicator that the physical or virtual processors are capable of simultaneous multi-threading.
SMT Mode	Current runtime simultaneous multi-threading mode of `disabled` or `enabled`.
SMT Boot Mode	Current boot time simultaneous multi-threading mode of `disabled` or `enabled`.
SMT Threads	Number of simultaneous multi-threading threads per physical or virtual processor.
SMT Bound	Indicator that the simultaneous multi-threading threads are bound on the same physical or virtual processor.

Exit Status

0	Successfully completed the requested operation.
>0	An error occurred.

Examples

1. To enable simultaneous multi-threading for the current boot cycle, type:

   ```
   smtctl -m on -w now
   ```

   The system displays a message similar to the following:

   ```
   smtctl: SMT is now enabled.
   ```

2. To view the current simultaneous multi-threading mode settings and processor information, type:

   ```
   smtctl
   ```

   The system displays a message similar to the following:

   ```
   This system is SMT capable.

   SMT is currently enabled.

   SMT boot mode is set to disabled.
   ```

```
proc0 has 2 SMT threads
Bind processor 0 is bound with proc0
Bind processor 1 is bound with proc0


proc2 has 2 SMT threads
Bind processor 2 is bound with proc2
Bind processor 3 is bound with proc2
```

3. To disable simultaneous multi-threading for the current boot cycle and for all subsequent boots, type:

```
smtctl -m off
```

The system displays a message similar to the following:

```
smtctl: SMT is now disabled. It will persist across reboots if
        you run the bosboot command before the next reboot.
```

Note: The boot image must be remade with the **bosboot** command before the next reboot.

Location

/usr/sbin/smtctl

Files

/usr/sbin/smtctl	Contains the **smtctl** command.

Related Information

The **bosboot**, **bindprocessor** command in *AIX 5L Version 5.3 Commands Reference, Volume 1*.

---

smundefca Command

Purpose

Unconfigures internal Certificate Authority.

Syntax

smundefca

Description

The **smundefca** command is used to unconfigure the internal certificate authority (CA) that was previously defined on the current machine.

The **smundefca** command removes the following files:

/usr/websm/security/SM.caprivkr	The CA private key ring which includes the CA private key and the CA certificate.
/usr/websm/security/SMCa.sn	The certificate number file.

The log file **/usr/websm/security/SMCa.log** is not deleted.

You can use Web-based System Manager (wsm) command to access the graphical interface.

Examples

```
smundefca
```

Files

/usr/websm/security/SMCa.log	Lists detailed information on all operations executed by the CA.
/usr/websm/security/SM.caprivkr	Certificate private key ring file.
/usr/websm/security/SMCa.sn	Certificate number file.

Related Information

The **smcaprop**, **smdefca**, **smexpcacert**, **smimpcacert**, **smlistcerts**, and the **smsigncert** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

---

snap Command

Purpose

Gathers system configuration information.

Syntax

snap [**-a**] [**-A**] [**-b**] [**-B**] [**-c**] [**-C**] [**-D**] [**-f**] [**-g**] [**-G**] [**-i**] [**-k**] [**-l**] [**-L**][**-n**] [**-N**] [**-p**] [**-r**] [**-R**] [**-s**] [**-S**] [**-t**] [**-T** *Filename*] [**-w**] [**-o** *OutputDevice*] [**-d** *Dir*] [**-v** *Component*] [**-O** *FileSplitSize*] [**-P** *Files*] [*script1 script2* ... | All | file:*filepath*]

snap [**-a**] [**-A**] [**-b**] [**-B**] [**-c**] [**-C**] [**-D**] [**-f**] [**-g**] [**-G**] [**-i**] [**-k**] [**-l**] [**-L**][**-n**] [**-N**] [**-p**] [**-r**] [**-R**] [**-s**] [**-S**] [**-t**] [**-T** *Filename*] [**-o** *OutputDevice*] [**-d** *Dir*] [**-v** *Component*] [**-O** *FileSplitSize*] [**-P** *Files*] [*script1 script2* ... | All | file:*filepath*]

snap -e [**-m** *Nodelist*] [**-d** *Dir*]

Description

The **snap** command gathers system configuration information and compresses the information into a **pax** file. The file may then be written to a device such as tape or DVD, or transmitted to a remote system. The information gathered with the **snap** command might be required to identify and resolve system problems.

Note: Root user authority is required to execute the **snap** command. Use the **snap -o /dev/cd0** command to copy the compressed image to DVD. Use the **snap -o /dev/rmt0** command to copy the image to tape.

Use the **snap -o /dev/rfd0** command to copy the compressed image to diskette. Use the **snap -o /dev/rmt0** command to copy the image to tape.

Approximately 8MB of temporary disk space is required to collect all system information, including contents of the error log. If you do not gather all system information with the **snap -a** command, less disk space may be required (depending on the options selected).

Note: If you intend to use a tape to send a snap image to IBM for software support, the tape must be one of the following formats:

- 8mm, 2.3 Gb capacity
- 8mm, 5.0 Gb capacity
- 4mm, 4.0 Gb capacity

Using other formats prevents or delays IBM software support from being able to examine the contents.

The **snap -g** command gathers general system information, including the following:

- Error report
- Copy of the customized Object Data Manager (ODM) database
- Trace file
- User environment
- Amount of physical memory and paging space
- Device and attribute information
- Security user information

The output of the **snap -g** command is written to the **/tmp/ibmsupt/general/general.snap** file.

The **snap** command checks for available space in the **/tmp/ibmsupt** directory, the default directory for **snap** command output. You can write the output to another directory by using the **-d** flag. If there is not enough space to hold the **snap** command output, you must expand the file system.

Each execution of the **snap** command appends information to previously created files. Use the **-r** flag to remove previously gathered and saved information.

Flags

-a Gathers all system configuration information. This option requires approximately 8MB of temporary disk space.

-A Gathers asynchronous (TTY) information.

-b Gathers SSA information.

-B Bypasses collection of SSA adapter dumps. The **-B** flag only works when the **-b** flag is also specified; otherwise, the **-B** flag is ignored.

-c Creates a compressed **pax** image (**snap.pax.Z** file) of all files in the **/tmp/ibmsupt** directory tree or other named output directory.
Note: Information not gathered with this option should be copied to the **snap** directory tree before using the **-c** flag. If a test case is needed to demonstrate the system problem, copy the test case to the **/tmp/ibmsupt/testcase** directory before compressing the **pax** file.

-C Retrieves all the files in the fwdump_dir directory. The files are placed in the ″general″ subdirectory. The **-C** snap option behaves the same as **-P***.

-D Gathers dump and **/unix** information. The primary dump device is used.

Notes:

1. If **bosboot -k** was used to specify the running kernel to be other than /**unix**, the incorrect kernel is gathered. Make sure that /**unix** is , or is linked to, the kernel in use when the dump was taken.
2. If the dump file is copied to the host machine, the **snap** command does not collect the dump image in the **/tmp/ibmsupt/dump** directory. Instead, it creates a link in the dump directory to the actual dump image.

-d *AbsolutePath* Identifies the optional **snap** command output directory (**/tmp/ibmsupt** is the default). You must specify the absolute path.

-e Gathers HACMP™ specific information.
Note: HACMP specific data is collected from all nodes belonging to the cluster . This flag cannot be used with any other flags except **-m** and **-d**.

-f Gathers file system information.

-g Gathers the output of the **lslpp -hac** command, which is required to recreate exact operating system environments. Writes output to the **/tmp/ibmsupt/general/lslpp.hBc** file. Also collects general system information and writes the output to the **/tmp/ibmsupt/general/general.snap** file.

-G	Includes predefined Object Data Manager (ODM) files in general information collected with the **-g** flag.
-i	Gathers installation debug vital product data (VPD) information.
-k	Gathers kernel information
-l	Gathers programming language information.
-L	Gathers LVM information.
-m *Nodelist*	Node name list (separated by commas) to gather HACMP information. **Note:** Currently this flag is only valid with the **-e** flag.
-n	Gathers Network File System (NFS) information.
-N	Suppresses the check for free space required.
-o *OutputDevice*	Copies the compressed image onto the specified device.
-O *FileSplitSize*	Used to enable splitting of the **snap** output files into smaller files. The size of these files is specified as a parameter to the **-O** option and must be specified in megabytes. This flag can only be used when the **-c** flag is specified.
-p	Gathers printer information.
-P *Files*	Retrieves the named *Files* from the **fwdump_dir** directory. If **-P *** is specified, all the files in the directory are gathered. The files are placed in the **general** subdirectory. The **-C snap** option behaves the same as **-P***.
-r	Removes **snap** command output from the **/tmp/ibmsupt** directory.
-R	Gathers SCSI RAID information.
-s	Gathers Systems Network Architecture (SNA) information.
-S	Includes security files in general information collected with the **-g** flag.
-t	Gathers Transmission control protocol information.
-T *Filename*	Gathers all the log files for a multi-CPU trace. Only the base file, **trcfile**, is captured with the **-g** flag.
-v *Component*	Displays the output of the commands executed by the **snap** command. Use this flag to view the specified name or group of files. **Note:** Press the Ctrl-C key sequence to interrupt the **snap** command. A prompt will return with the following options: press the Enter key to return to current operation; press the S key to stop the current operation; press the Q key to quit the **snap** command completely.
-w	Gathers WLM information.

Parameters

Arguments

Names of third-party scripts to be executed are specified as parameters to **snap**. A parameter can be a single word or a list of words enclosed in quotes. When parameters are enclosed in quotes, the first parameter in the list represents the name of the script and the subsequent words represent the arguments to pass to the script.

When **All** is specified as a parameter, all the scripts in the script repository are executed. No script parameters may be passed in this case.

If the **file:** keyword is used and is immediately followed by a path to a file, that file is read to get the scripts to execute. Each line in the file represents a script and optional parameters to the script .

snap Scripts

A third-party script must be executable in **/usr/lib/ras/snapscripts**, and must follow the guidelines described below. When called during pass 1, a script must return its size estimation to **snap**. In pass 2, it collects the data and saves it as specified by **snap**.

The script must read and utilize the following environment variables, SNAPDIR, PASSNO, SCRIPTSIZE and SCRIPTLOG.

All output files must be written to $SNAPDIR. This is the directory where the script should be saving its output. The PASSNO variable contains the **snap** phase during which the script is called. During the first

pass, the script should calculate a size estimation for the data it will write during the 2nd pass. It will then write that numeric estimation to the file pointed to by $SCRIPTSIZE. The value saved to the file should be in decimal. **snap** passes the path to a log file where all debug data for the script should be saved. Standard out and standard error should not be redirected by the script, because **Snap** will save standard out and standard error to $SNAPDIR/*ScriptName*.**out** and $SNAPDIR/*ScriptName*.**err**, respectively.

Example:

```
#!/usr/bin/ksh


if [ "$PASSNO" = 1 ]
then
         (( size=99999 ))
        ....
        # this is where code to do the size estimation should go.
        ....
        echo $size > $SCRIPTSIZE

else if [ "$PASSNO" = 2 ]
then
        # debug information should go to $SCRIPTLOG
        echo "Debug Data" >> $SCRIPTLOG

        # .....where the work to collect the data takes place
        # ...

        # The data collected should be written to $SNAPDIR        .
        touch $SNAPDIR/foo_output1
        touch $SNAPDIR/foo_output2
fi
fi
```

Note: To collect information about virtual SCSI devices run the command **snap client_collect,all**.

Splitting of Snap Output: If it is split, **snap** output might look like the following:

```
% ls -l
total 112048
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaa
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xab
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xac
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xad
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xae
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaf
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xag
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xah
-rw-r--r--   1 lmic     adm         6291456 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xai
-rw-r--r--   1 lmic     adm          744518 Nov 26 09:56 snap.hastings.112603095649.pax.Z.xaj
```

Examples

1. Enter the following command to gather all system configuration information:

   ```
   snap -a
   ```

   The output of this command is written to the **/tmp/ibmsupt** directory.
2. Enter the following command to create a **pax** image of all files contained in the **/tmp/ibmsupt** directory:

   ```
   snap -c
   ```
3. Enter the following command to gather general system configuration information, including the output of the **lslpp -hBc** command:

```
snap -g -o /dev/rfd0
```

Output is written to the **/tmp/ibmsupt/general/lslpp.hBc** and **/tmp/ibmsupt/general/general.snap** files. This command also writes the system information to a removable diskette.

4. Enter the following command to gather HACMP specific information from nodes `node1` and `node2` belonging to a single cluster:

```
snap -e -m node1,node2
```

Output is written to the **/tmp/ibmsupt/hacmp** directory.

5. To run the scripts foo1, foo2 and foo3. where foo1 takes no argument, foo2 takes three arguments and foo3 takes one argument, type the following:

```
snap  foo1 "foo2 -x -y 3" "foo3 6"
```

Output is written to **/tmp/ibmsupt/snapscripts/foo1**, **/tmp/ibmsupt/snapscripts/foo2** and **/tmp/ibmsupt/snapscripts/foo3** assuming the destination directory is the default, **/tmp/ibmsupt**.

6. To specify the **All** parameter to run all the scripts, type:

```
snap All
```

Note: No parameters are passed in this case.

7. To specify the path to a file containing the name and optional parameter list of scripts to execute, type:

```
snap file:/tmp/scriptnames
```

A sample input file to execute the scripts from example 5:

```
foo1
foo2 -x -y 3
foo6
```

8. If splitting of the **snap** output into 4MB files is desired, type:

```
snap -a -c -O 4
```

Files

/usr/sbin/snap	Contains the **snap** command.
/tmp/ibmsupt	Contains **snap** command output.
/tmp/ibmsupt/general/lslpp.hBc	Contains the output of the **lslpp -hBc** command, which is required to recreate exact operating system environments.
/tmp/ibmsupt/general/general.snap	Contains general system information that is collected with the **snap -g** command.
/tmp/ibmsupt/testcase	Contains the test case that demonstrates your system problem.

Related Information

The **errpt** command, **gettrc** command, **lslpp** command, **snapsplit** command, **sysdumpdev** command, **sysdumpstart** command.

---

snapcore Command

Purpose

Gathers the **core** file.

Syntax

snapcore[**-d** *Dir*] [**-r**] core [*program*]

Description

The **snapcore** command gathers the **core** file, program, and libraries used by the program and compresses the information into a **pax** file. The file can then be downloaded to disk or tape, or transmitted to a remote system. The information gathered with the **snapcore** command is required to identify and resolve a problem with the application.

The **snapcore** command checks for available space in the **/tmp/snapcore** directory, the default directory for **snapcore** command output. You can write the output to another directory by using the **-d** flag. If there is not enough space to hold the **snapcore** command output, you must expand the file system.

Each execution of the **snapcore** command creates a new archive file. The archive file is named **snapcore_$pid.pax**. Use the **-r** flag to remove the previously created archive file. This command uses **$pid** (pid of the **snapcore** command) to create a unique name file and preserve any previously created archives.

Specify the full path name for core and program. If the program name is not specified, **snapcore** reads the program name from the **core** file and searches for the location in directories contained in the *PATH* variable.

Flags

-d*Dir* Identifies the optional **snapcore** command output directory (**/tmp/snapcore** is the default).
-r Removes **snapcore** command output from the **/tmp/snapcore** directory.

Examples

1. To gather the **core** file, enter the following:
   a. `snapcore <core file name> <program name>`
   b. `snapcore <core file name>`

   Directories contained in the *PATH* variable are searched to find the program file. The **pax** file is created in **/tmp/snapcore** directory.
2. To clean the previously created core archive and create a new one, enter the following:

   ```
   snapcore -r<core file name> <program name>
   ```

   The **pax** file is created in **/tmp/snapcore** directory.
3. To create the **core** file archive in an alternate directory, enter the following:

   ```
   snapcore -d<dir name> <core file name> <program name>
   ```

   The **pax** file is created in **<dirname>/tmp/snapcore** directory.
4. To clean the **/tmp/snapcore** directory, enter the following:

   ```
   snapcore -r
   ```

Files

/usr/sbin/snapcore	Contains the **snapcore** command.
/tmp/snapcore	Contains **core** file archive.

Related Information

The **dbx** command, **pax** command.

snapshot Command

Purpose

Modify, create or view properties of snapshots.

Syntax

snapshot { **-o snapfrom=***snappedFS* **-o size=***Size* | {**-d** [**-s**] | **-q** [**-c***fieldSeparator*] | **-o snapfrom=***snappedFS* | **-o size=***Size*} *Object*}

Description

Provides an interface to JFS2 snapshots.

Flags

-d	Deletes the snapshot and the logical volume containing the snapshot, along with any prior snapshots. The *Object* specified is a snapshot storage object.
-s	Retains the specified logical volume for the specified snapshot when the snapshot is deleted.
-q	Displays information about the specified snapshot if the specified *Object* is a snapshot storage object. Displays the file system the snapshot belongs to, the time of when the snapshot was taken, the size of the snapshot storage object and the remaining free space. If the Object specified is a snappedFS, display information about all snapshots for the snappedFS. Show each of the snapshots and their storage object along with the time when the snapshot was taken. Displays the size of the snapshot storage objects and the remaining free space.
-c*fieldSeparator*	Specifies the output from the snapshot query should be displayed in colon format. The *fieldSeparator* is the character to use to separate the fields of the display.
-o snapfrom=*snappedFS*	Creates a snapshot of the specified *snappedFS*. If the *Object* is not specified, a new logical volume will be created and used for the snapshot. If the specified *Object* is a logical volume, it must already exist and it must be in the same volume group as the *snappedFS*. If the Object specified is already in use as a snapshot or a file system known to the **/etc/filesystems** file, the command issues an error message and fails.
-o size=*Size*	Specifies the size of a new logical volume if specified along with the **-o snapfrom=***snappedFS* flag. Otherwise, this flag increases the size of the snapshot specified by the *Object* field to the value of *Size*. This flag is Ignored if any other flag is given. If *Size* is followed by an M the value is treated as megabytes. If *Size* is followed by a G the value is treated as gigabytes. If neither M or G are used, the value is treated as 512-byte blocks.

Parameters

Object	Use depends on other flags specified, see flag description for interpretation of Object. This may be a device or a file system.

Examples

1. To create a snapshot for the **/home/janet/sb** file system on the **/dev/snapsb** logical volume, type the following:

   ```
   snapshot -o snapfrom=/home/janet/sb /dev/snapsb
   ```

   This command creates a snapshot for the **/home/janet/sb** file system on the **/dev/snapsb** logical volume, which already exists.
2. To create a snapshot for the **/home/janet/sb** file system, type the following:

   ```
   snapshot -o snapfrom=/home/janet/sb -o size=16M
   ```

   This command creates a 16-megabyte logical volume and creates a snapshot for the **/home/janet/sb** file system on the newly created logical volume.
3. To view information about all the snapshots for the **/home/janet/sb** file system, type the following:

   ```
   snapshot -q /home/janet/sb
   ```

   This command displays each snapshot for the **/home/janet/sb** file system along with the time when the snapshot was taken, the size of the snapshot storage object, and the remaining free space.
4. To increase the size of the snapshot on the **/dev/snapsb** device, type the following:

   ```
   snapshot -o size=64M /dev/snapsb
   ```

   This command increases the **/dev/snapsb** device to 64 megabytes along with the snapshot contained on the device.
5. To delete the snapshot on the **/dev/snapsb**device, type the following:

   ```
   snapshot -d /dev/snapsb
   ```

   This command deletes the snapshot contained on the **/dev/snapsb** device and removes the **/dev/snapsb** logical volume .

---

snapsplit Command

Purpose

To split a **snap** output file into multiple smaller files of arbitrary or specified size.

Syntax

snapsplit [**-s** *size*] [**-H** *machinename*] [**-f** *filename*]

snapsplit -u -T *timestamp* [**-H** *machinename*]

Description

The **snapsplit** command is used to split a **snap** output file into smaller files. This command is useful for dealing with very large **snap** files. It breaks the file down into files of a specific size that are multiples of 1 megabyte. Furthermore, it will combine these files into the original file when called with the **-u** option.

The output files are named as following: **snap.*machinename.timestamp*.pax.Z.xxx**. *Machinename* is the hostname and *timestamp* is in the format MMDDYYHHMMSS. In addition, xxx represents the extension for

the **split** files which is crucial when putting these files back together. The extensions from the start of the files go in the following order: **xaa, xab, xac, xad, xae ..., xaz, xba, xbb, xbc, xbd, ..., xbz, xca, xcb, xcc, ...**.

When performing **ls** on these files, the first file listed would represent the top of the original file and the last file, the end of the original file.

Note that this command should only be used for **snap** files that are **paxed** and compressed. When executed on local system where **snap** output was gathered, the **-H** option need not be used. That flag is provided for the case where user has moved a complete **snap** file to a remote system and wishes to split it. Any machine name may be selected, but it is recommended, to use the machine name where data was collected.

Flags

-f *filename*	Input **snapsplit** file. It should be a compressed **pax** file. The default is **snap.pax.Z**.
-H *machinename*	Name of the host machine. If none is specified, the default is the current host. Care must be exercised to name **snap** files for the appropriate system.
-s *size*	Specifies the size of **snap** output in multiples of 1 MB. The last file will be smaller or equal to this size. *Size* should be entered in megabytes. The default size is 1 MB.
-T *timestamp*	Timestamp of the **snapsplit** files to use in restoring the original **snap** output. It is in the format MMDDYYHHMMSS, where MM for month, DD for day, YY for year, HH for hours, MM is for minutes and SS is for seconds.
-u	Flag used for rejoining **snapsplit** files. Used with the **-T** flag.

Examples

1. To split the default snap file (**snap.pax.Z** should be in the current directory), enter the following:

   ```
   snapsplit
   ```

   The output of this command is written to current directory.
2. To split file **snap.somefile.pax.Z** from system doe, enter the following:

   ```
   snapsplit -H doe -f snap.somefile.pax.Z
   ```

   Note: The resulting files will be named **snap.doe.MMDDYYHHMMSS.pax.Z**.
3. To restore a file for which the **snap** files (**snap.sue.102303141211.xxx**) are for system sue, and timestamp 102303141211, type:

   ```
   snapsplit -u -T 102303141211 -H sue
   ```

   Attention: If any one of the **snap** files is missing or has been renamed, the **snap** file created will corrupted.
4. To restore a **snap** file from files with time stamp 102603084512, and which are for the current system, type:

   ```
   snapsplit -u -T 102603084512
   ```
5. To gather general system configuration information, including the output of the **lslpp -hBc** command, type the following:

   ```
   snap -g -o /dev/rfd0
   ```

   Output is written to the **/tmp/ibmsupt/general/lslpp.hBc** and **/tmp/ibmsupt/general/general.snap** files. This command also writes the system information to a removable diskette.

Files

/usr/sbin/snapsplit	Contains the **snapsplit** command.

Related Information

The **snap**, **split**, and **cat** commands.

---

snmpevent Command

Purpose

Sends ERRM events to an SNMP manager.

Syntax

snmpevent [**-a** *host-name*] [**-c** *community*] [**-h**]

Description

The **snmpevent** script sends a Simple Network Management Protocol (SNMP) trap of an event response resource manager (ERRM) event to a host running an SNMP agent. The agent formats the trap information into an SNMP trap and sends it to the SNMP manager defined in its configuration file. This script is meant to be called by the predefined ERRM response **Generate SNMP trap**. Event or rearm event information is captured and posted by ERRM in environment variables that are generated when an ERRM event or a rearm event occurs.

The **snmpevent** script can also be used as a template to create other user-defined actions. See the *RSCT Administration Guide* to understand how an event response resource runs an action command.

The following message template is sent as a trap when an event or a rearm event occurs and **snmpevent** is the defined response:

```
[ERRM_COND_SEVERITY] [ERRM_TYPE] occurred:
          Condition: [ ERRM_COND_NAME]
          Node:  [ERRM_NODE_NAME]
          Resource: [ERRM_RSRC_NAME]
          Resource Class:  [ERRM_RSRC_CLASS_NAME]
          Resource Attribute: [ERRM_ATTR_NAME]
          Attribute Type: [ERRM_DATA_TYPE]
          Attribute Value: [ERRM_VALUE]
```

The environment variables have the following definitions:

ERRM_COND_SEVERITY
: Specifies the significance of the condition resource that caused the event or rearm event. The valid values are: Critical, Warning, or Informational.

ERRM_TYPE
: Specifies the type of event that occurred. The valid values are: event or rearm event.

ERRM_COND_NAME
: Specifies the name of the condition resource with the attribute value that changed to cause this event or rearm event.

ERRM_NODE_NAME
: Specifies the host name on which this event or rearm event occurred.

ERRM_RSRC_NAME
: Specifies the name of the resource with the attribute that changed to cause this event or rearm event.

ERRM_RSRC_CLASS_NAME
Specifies the name of the resource class to which the resource that caused this event or rearm event belongs.

ERRM_ATTR_NAME
Specifies the name of the resource attribute that changed to cause this event or rearm event.

ERRM_DATA_TYPE
Specifies the data type of the resource attribute.

ERRM_VALUE
Specifies the value of the resource attribute that changed to cause this event or rearm event.

The **snmpevent** command captures these environment variable values and formats a generic message that is sent as a trap via a call to the **snmptrap** command.

Flags

–a *host-name*
Specifies the host name of the SNMP agent to which the AIX subagent will connect. By default, the subagent will connect to the SNMP agent running on the local node.

–c Specifies the SNMP community to be used. This can be any string the SNMP agent will accept. The default is **public**.

–h Writes this script's usage statement to standard output.

Parameters

log_file
Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 The script has run successfully.

1 An error occurred when the script was run.

Restrictions

This script must be run on the node where the ERRM is running.

Standard Output

When the **-h** flag is specified, this script's usage statement is written to standard output.

Examples

1. Suppose the command **/usr/sbin/rsct/bin/snmpevent** is an action in the critical-notification response, which is associated with the CSM predefined condition **NodeChanged**. This can be done with the **mkcondresp** command followed by the **startcondresp** command. The **/etc/snmpdv3.conf** file should be configured to where the trap will be sent. In this example, if you want the trap sent to **9.117.16.246**, write the **/etc/snmpdv3.conf** file as follows:

```
VACM_GROUP group1 SNMPv1  public  -

VACM_VIEW defaultView        internet                    - included
-VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -

NOTIFY notify1 traptag trap -

#TARGET_ADDRESS Target1 UDP 127.0.0.1       traptag trapparms1 - - -
TARGET_ADDRESS Target1 UDP 9.117.16.246       traptag trapparms1 - - -

TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -

COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -

DEFAULT_SECURITY no-access - -

logging         file=/usr/tmp/snmpdv3.log       enabled
logging         size=0                          level=0

smux            1.3.6.1.4.1.2.3.1.2.1.2         gated_password  # gated

snmpd smuxtimeout=200 #muxatmd
smux 1.3.6.1.4.1.2.3.1.2.3.1.1 muxatmd_password #muxatmd
```

Then, restart the **snmpd** daemon by first killing the **snmpd** daemon that is currently running and then starting it again:

```
# ps -ef | grep snmpd
    root  4570 12956   1 08:24:32  pts/0  0:00 grep snmpd
    root 13810     1   0 08:11:04      -  0:00 snmpd
# kill -9 13810
# snmpd
```

Next, change the LParID property of node c175n08 to 12:

```
# chnode c175n08 LParID=12
```

Now, on the node **9.117.16.158** (the node with the SNMP manager that was specified in the **/etc/snmpdv3.conf** file), the SNMP manager should record something like this:

```
2002-07-15 09:09:25 c174tr1.ppd.pok.ibm.com [9.114.78.17] TRAP, SNMP v1,
community public
        enterprises.ibm Enterprise Specific Trap (1) Uptime: 0:01:45.00
        enterprises.ibm.ibmProd.191.1.6.1.0 = "Informational Event
occurred. Condition=NodeChanged Node=c174tr1.ppd.pok.ibm.com
Resource=c175n08.ppd.pok.ibm.com Resource Class=Node Resource
Attribute=Changed Attributes Attribute Type=CT_CHAR_PTR_ARRAY Attribute
Val={LParID} "
```

The output varies based on SNMP managers.

Location

/usr/sbin/rsct/bin/snmpevent Contains the **snmpevent** script

Related Information

Commands: **lscondition**, **lscondresp**, **lsresponse**, **mkcondition**, **mkcondresp**, **mkresponse**, **snmptrap**, **startcondresp**, **stopcondresp**

Books: see the *RSCT Administration Guide* for information about the event response resource manager (ERRM) and about how to use ERRM environment variables

snmpd Daemon

Purpose
Starts the Simple Network Management Protocol (SNMP) agent as a background process.

Syntax
Refer to the syntax for either the **snmpdv1** daemon or the **snmpdv3** daemon.

Description
/usr/sbin/snmpd is a symbolic link to either the **snmpdv1** daemon which supports only SNMP version 1, or to either the encrypted or non-encrypted version of the **snmpdv3** daemon which supports SNMP version 3. For information about the **snmpd** daemon, please refer to the documentation of the appropriate version of the SNMP agent. On AIX 5.2, the default SNMP agent running at system boot time is the non-encrypted version of the SNMP version 3 agent.

Note: The encrypted version of the SNMP version 3 agent is available from the AIX Expansion Pack.

Files

/usr/sbin/snmpd	Contains a symbolic link to either **/usr/sbin/snmpdv1**, **/usr/sbin/snmpdv3e**, or **/usr/sbin/snmpdv3ne**.
/usr/sbin/snmpdv1	Contains the SNMP version 1 agent.
/usr/sbin/snmpdv3e	Contains the encrypted version of the SNMP version 3 agent.
/usr/sbin/snmpdv3ne	Contains the non-encrypted version of the SNMP version 3 agent.

Related Information
The **snmpv3_ssw** command.

The SNMP for network management chapter in *Networks and communication management*.

snmpdv1 Daemon

Purpose
Starts the Simple Network Management Protocol (SNMP) version 1 agent as a background process.

Syntax
snmpd [**-c** *ConfigFile*] [**-d** *Level*] [**-f** *LogFile*] [**-S**]

Description
The **snmpd** command starts the SNMP daemon. This command may only be issued by a user with root privileges or by a member of the system group.

The SNMP daemon is a server that supports the standard Simple Network Management Protocol (SNMP) documented by RFC 1157 and the Management Information Base (MIB) as defined in RFC 1155 and RFC 1213. The SNMP daemon provides the following three functions:

- Receiving and authenticating SNMP requests from network monitors.
- Processing requests and returning results to the originating monitor.
- Sending trap notification to all hosts listed in the configuration file.

The SNMP daemon server keeps log messages in a file specified by the *LogFile* variable if the **-f** flag is used or in a log file specified in the configuration file. When the size of the log file exceeds the predefined maximum log file size, the **snmpd** command will rotate the log file by moving the old log file to another file as follows:

- LogFile.3 is deleted.
- LogFile.2 is moved to LogFile.3.
- LogFile.1 is moved to LogFile.2.
- LogFile.0 is moved to LogFile.1.
- LogFile is moved to LogFile.0.
- Logging continues in LogFile.

If logging is not directed from the **snmpd** command line with the **-f** flag, logging can be directed from the configuration file.

Supported set variables are:

- **sysContact**
- **sysName**
- **sysLocation**
- **ifAdminStatus**
- **atPhysAddress**
- **atNetAddress**
- **ipForwarding**
- **ipDefaultTTL**
- **ipRouteDest**
- **ipRouteNextHop**
- **ipRouteType**
- **ipNetToMediaPhysAddress**
- **ipNetToMediaNetAddress**
- **ipNetToMediaType**
- **snmpEnableAuthenTraps**
- **smuxPstatus**
- **smuxTstatus**

See ″Understanding SNMP Daemon Support for SET Request Processing″ in *AIX 5L Version 5.3 Communications Programming Concepts* for more information on the supported set variables.

The following commands should be issued before the SNMP daemon is started:

- **ifconfig lo0 loopback**
- **startsrc -s inetd**

These commands are normally executed during system startup when the **/etc/rc.net** and **/etc/rc.tcpip** shell scripts are called. (The **snmpd** command can be placed in the **/etc/rc.tcpip** shell script.)

The **snmpd** daemon should be controlled using the System Resource Controller (SRC). Entering **snmpd** at the command line is not recommended.

Manipulating the snmpd Daemon with the System Resource Controller

The **snmpd** daemon is a subsystem controlled by the System Resource Controller (SRC). The **snmpd** daemon is a member of the **tcpip** system group. The **snmpd** daemon is enabled by default and can be manipulated by SRC commands.

Use the following SRC commands to manipulate the **snmpd** daemon:

startsrc	Starts a subsystem, group of subsystems, or a subserver. Issuing the **startsrc** command causes the **snmpd** command to generate a *coldStart* trap.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes a subsystem or group of subsystems to reread the appropriate configuration file. Issuing a **refresh** command causes the **snmpd** daemon to generate a *warmStart* trap.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver. If the user issuing the **traceson** command is not the root user, the debugging level will not exceed level 2.
tracesoff	Disables tracing of a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver. If the user issuing the long status form of the **lssrc** command is not the root user, no community name information is displayed.

Flags

-c *ConfigFile*	Specifies full path and file name of the configuration file for the **snmpd** daemon. This file is read when the **snmpd** daemon starts up and when a **refresh** or **kill -1** signal is issued. If the **-c** flag is not specified, the default configuration file is **/etc/snmpd.conf**. See the **snmpd.conf** file for information on this file format.
-d *Level*	Specifies the level of tracing the **snmpd** command produces. The *Level* value can be one of: **0** All notices, exceptions, and fatal messages **1** Level 0 plus debug messages **2** Level 1 plus a hexadecimal dump of incoming and outgoing packets **3** Level 2 plus an English version of the request and response packets If the **-d** flag is not specified, the debugging level is set to 0.
-f *LogFile*	Specifies the full path and file name into which **snmpd** tracing information is logged. If the **-f** flag is not specified, no information will be logged. See the **snmpd.conf** file for more information on setting logging parameters.
-S	Enable the security option if it's specified. It will prevent the local non-root user from changing the value of MIB variable(s) on the local host.

Examples

1. To start the **snmpd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpd -a "-f /tmp/snmpd.log"
   ```

   This command starts the **snmpd** daemon and logs information to the **/tmp/snmpd.log** file at debug level 0.
2. To stop the **snmpd** daemon normally, enter:

   ```
   stopsrc -s snmpd
   ```

   This command stops the daemon. The **-s** flag specifies the subsystem that follows to be stopped.
3. To get short status from the **snmpd** daemon, enter:

   ```
   lssrc -s snmpd
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To get a long status from the **snmpd** daemon, enter:

   ```
   lssrc -ls snmpd
   ```

   If you are the root user, this long form of the status report lists the configured community names and associated access privileges and views for **snmp** requests. The long form also lists the community

names associated with the hosts for trap notification, logging configuration parameters, **snmpd** specific configuration parameters and **smux** configuration parameters.

5. To enable tracing for the **snmpd** daemon, enter the following:

   ```
   traceson -s snmpd
   ```

   This command enables **snmpd** debugging if the **snmpd** daemon is configured for logging.
6. To view the contents of the DHCP Server database files **/etc/dhcpsd.ar** and **/etc/dhcpsd.cr**, enter:

   ```
   lssrc -l -s dhcpsd
   ```

Files

/etc/services Contains port assignments for required services. The following entries must be present in the **/etc/services** file if the entries are not already present:

snmp 161/udp

snmp-trap 162/udp

smux 199/tcp

Notes:

1. The snmp port *must* be 161 as required by RFC 1157.
2. The snmp-trap port *must* be 162 as required by RFC 1157.
3. The smux port *must* be 199.
4. The **/etc/services** file is shipped with these entries already in place.
5. If the **/etc/services** file is being served from a server, these entries must be present in the server's **/etc/services** file.

/etc/snmpd.conf Specifies the configuration parameters for the **snmpd** agent.

/etc/mib.defs Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle.

Related Information

The **gated** daemon, **snmpd** daemon, **snmpv3_ssw** command.

---

snmpdv3 Daemon

Purpose

Starts the Simple Network Management Protocol (SNMP) version 3 agent as a background process.

Syntax

snmpd [**-d** *Level*] [**-i** *Interval*] [**-p** *Port*] [**-S**]

Description

The **snmpd** command starts the SNMP daemon. This command may only be issued by a user with root privileges or by a member of the system group.

The SNMP daemon is a server that supports the all the SNMPv1, SNMPv2c, and SNMPv3 protocols documented by RFCs 1157, RFD 1905, and RFC 2572. It also behaves as a SMUX server as defined by RFC 1227 and as a DPI2 agent as defined by RFC 1592. The SNMP daemon provides the following three functions:

- Receiving and authenticating SNMP requests from network monitors.

- Processing requests and returning results to the originating monitor.
- Sending trap notification to all hosts listed in the configuration file.

The SNMP daemon server keeps log messages in a file specified by the *LogFile* variable if the **-f** flag is used or in a log file specified in the configuration file. When the size of the log file exceeds the predefined maximum log file size, the **snmpd** command will rotate the log file by moving the old log file to another file as follows:

- LogFile.3 is deleted.
- LogFile.2 is moved to LogFile.3.
- LogFile.1 is moved to LogFile.2.
- LogFile.0 is moved to LogFile.1.
- LogFile is moved to LogFile.0.
- Logging continues in LogFile.

The following commands should be issued before the SNMP daemon is started:

- **ifconfig lo0 loopback**
- **startsrc -s inetd**

These commands are normally executed during system startup when the **/etc/rc.net** and **/etc/rc.tcpip** shell scripts are called. (The **snmpd** command can be placed in the **/etc/rc.tcpip** shell script.)

The **snmpdv3** daemon should be controlled using the System Resource Controller (SRC). Entering **snmpd** at the command line is not recommended.

Manipulating the snmpd Daemon with the System Resource Controller

The **snmpdv3** daemon is a subsystem controlled by the System Resource Controller (SRC). The **snmpdv3** daemon is a member of the **tcpip** system group. The **snmpdv3** daemon is enabled by default and can be manipulated by SRC commands.

Use the following SRC commands to manipulate the **snmpd** daemon:

startsrc	Starts a subsystem, group of subsystems, or a subserver. Issuing the **startsrc** command causes the **snmpdv3** command to generate a *coldStart* trap.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

-d *Level*
Specifies the level of tracing to be started. The valid values for level are 0-255. If the **-d** parameter is not specified, then the default level of 0 is used, meaning no tracing will be done. If the **-d** parameter is specified without a level, then a level of 31 is used, meaning all SNMP requests/responses/traps and DPI® activity will be traced.

There are 8 levels of tracing provided. Each level selected has a corresponding number. The sum of the numbers associated with each level of tracing selected is the value which should be specified as level. The numbers for the trace levels are:

0 No tracing. This is the default.

1 Trace SNMP responses, requests, and traps.

2 Trace DPI level 1 and DPI level 2.

3 Same as level 1 plus level 2 plus internal trace.

4 Same as trace level 3 plus extended trace.

-i *Interval*
Specifies the interval (in minutes) at which dynamic configuration changes to the SNMP agent should be written out to the **/etc/snmpdv3.conf** configuration file. Valid values are 0-10. The default value is 5. This parameter is only relevant when the **/etc/snmpdv3.conf** file is used for SNMPv3 configuration.

-p *Port*
Listens for SNMP packets on this port. The default is port 161.

-S
Prevents non-root users from changing the MIB values.

Examples

1. To start the **snmpd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpd
   ```

   This command starts the **snmpd** daemon at debug level 0.
2. To stop the **snmpd** daemon normally, enter:

   ```
   stopsrc -s snmpd
   ```

   This command stops the daemon. The **-s** flag specifies the subsystem that follows to be stopped.
3. To get status from the **snmpd** daemon, enter:

   ```
   lssrc -s snmpd
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

Files

/etc/services
Contains port assignments for required services. The following entries must be present in the **/etc/services** file if the entries are not already present:

snmp 161/udp

snmp-trap
162/udp

smux 199/tcp

/etc/snmpdv3.conf	Specifies the configuration parameters for the **snmpdv3** agent.
/etc/snmpd.boots	Specifies the engineID and the engineBoots for the **snmpdv3** agent.
/etc/mib.defs	Defines the Management Information Base (MIB) variable the SNMP agent should recognize and handle.

Related Information

The **clsnmp** command, **pwchange** command, **pwtokey** command, **snmpd** daemon,**snmpv3_ssw** command.

The **/etc/clsnmp.conf** file.

The SNMP for network management chapter in *Networks and communication management*.

---

snmpinfo Command

Purpose

Requests or modifies values of Management Information Base (MIB) variables managed by a Simple Network Management Protocol (SNMP) agent.

Syntax

The get or next Option

snmpinfo [**-m get** | **next**] [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] *Variable*. *Instance* ...

The set Option

snmpinfo -m set [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] *Variable* . *Instance*= *Value* ...

The dump Option

snmpinfo -m dump [**-v**] [**-c** *Community*] [**-d** *Level*] [**-h** *HostName*] [**-o** *ObjectsFile*] ... [**-t** *Tries*] [**-w** *Waittime*] [*Variable***.** *Instance*] ...

Description

The **snmpinfo** command requests or modifies values for one or more MIB variables for an SNMP agent. This command may only be issued by a user with root privileges or by a member of the system group.

If the you specify the **get** option, the **snmpinfo** command requests information about one or more MIB variables from an SNMP agent.

If you specify the **next** option, the **snmpinfo** command requests information from an SNMP agent about the instances following the specified instances. The **next** option makes it possible to obtain MIB values without knowledge of the instance qualifiers.

If you specify the **set** option, the **snmpinfo** command modifies values for one or more MIB variables for an SNMP agent. Only a few MIB variables are designated read-write. The agent that manages the MIB database may take various actions as a side effect of modifying MIB variables. For example, setting the **ifAdminStatus** MIB variable to 2 will normally shut down a network interface. The action taken is determined by the implementation of the SNMP agent that manages the database.

If you specify the **dump** option, the **snmpinfo** command can be used to traverse the entire MIB tree of a given agent. If a group is passed in as the *Variable* parameter, the **snmpinfo** command will traverse that specified path of the MIB tree.

The **snmpinfo** command has a debug facility that will dump debug information for transmitted and received packets. The facility is enabled with the **-d** flag.

Parameters

Value Specifies the value to which the MIB *Variable* parameter is to be set. A value must be specified for each variable. If a value is not specified, the request packet will be invalid.

Variable Specifies the name in text format or numeric format of a specific MIB variable as defined in the **/etc/mib.defs** file. If the option to the **-m** flag is **next** or **dump**, the *Variable* parameter may be specified as a MIB group.

Instance Specifies the instance qualifier for the MIB *Variable* parameter. The *Instance* parameter is required if the option to the **-m** flag is **get** or **set**. The *Instance* parameter is optional if the option to the **-m** flag is **next** or **dump**.

Notes:

1. There should be no blank spaces in the *Variable.Instance* parameter sequence.
2. If the *Instance* parameter is not specified, do not place a . (dot) after the *Variable* parameter.

For further information, consult RFC 1213, which defines the Management Information Base (MIB) for network management, and RFC 1157, which defines the SNMP protocol for creating requests for MIB information and formatting responses.

Flags

-c *Community* Specifies the community name to be used to query the SNMP agent. If the **-c** flag is not specified, the default community name is **public**.

-d *Level* Specifies the level of I/O debug information. The *Level* value can be one of:

- **0** No debug information.
- **1** Port bindings and the number of bytes transmitted and received.
- **2** Level 1 plus a hexadecimal dump of incoming and outgoing packets.
- **3** Level 2 plus an English version of the request and response packets.

If the **-d** flag is not specified, the default debug level is 0.

-h *HostName* Specifies the host name of the SNMP agent to be queried. If the **-h** flag is not specified, the default host name is the host name of the machine on which the user is currently logged in.

-m *Option* Specifies the mode by which to access the MIB variables.

The *Option* value can be one of:

get Requests information about the specified MIB variables.

next Requests the instances following the specified instances.

set Modifies the specified write access MIB variables.

dump Dumps the specified section of the MIB tree.

Notes:

1. The option name can be specified by the minimum number of characters required to make it unique.
2. If the **-m** flag is not specified, the default mode is **get**.

-o *ObjectsFile* Specifies the name of the objects definition file that defines the MIB objects the **snmpinfo** command can request. If the **-o** flag is not specified, the default objects definition file name is **/etc/mib.defs**. See the **mosy** command for information on creating this file. More than one *ObjectsFile* can be referenced with the restriction that files containing parent definitions be specified before files containing child definitions.

-t *Tries* Specifies the number of times the **snmpinfo** command transmits the SNMP request to the SNMP agent before terminating with the message `no SNMP response`. If the **-t** flag is not specified, the default number of tries is 3.

-v Specifies that the output from the **snmpinfo** command be displayed in verbose mode. If the **-v** flag is not specified, the information will not be displayed in verbose mode.

-w Specifies the wait time in seconds for the response from the **snmpd** agent. If the **-w** flag is not specified, the default wait time is 15 seconds.

Examples

1. To get the values for the MIB variable `ifDescr.1`, for the interface associated with `ifIndex.1` and `SysDescr`, enter:

   ```
   snmpinfo -m get -v sysDescr.0 ifDescr.1
   ```

   In this example, the **-m get** flag specifies that the **snmpinfo** command should retrieve the value of MIB variables `ifDescr.1`, (the interface description for the interface associated with the `ifIndex.1`)**,** and `sysDescr.0` (the system description of the local host).

2. To get the value for the MIB variable following the **ipAdEntIfIndex** MIB variable for the host specified by IP address 192.100.154.1, enter:

   ```
   snmpinfo -m next -v 1.3.6.1.2.1.4.20.1.2.192.100.154.1
   ```

   In this example, the **-m next** flag specifies that the **snmpinfo** command should retrieve the information for the MIB variable **ifAdEntIfIndex.**`192.100.154.1.`

3. To get the value of the first MIB variable in the system group, enter:

   ```
   snmpinfo -m next -v -h giants system
   ```

   In this example, the **-m next** flag specifies that the **snmpinfo** command should retrieve the information for the MIB variable following the system group, which is `sysDescr.0`; the **-v** flag indicates verbose mode; the **-h** flag indicates that the agent to be queried is `giants`**;** the group to retrieve information from is `system`.

4. To set the value of a MIB variable, enter a command similar to the following:

   ```
   snmpinfo -m set -v -h giants -c monitor -t 2 ifAdminStatus.1=2
   ```

   In this example, the MIB **ifAdminStatus** variable is set to 2, or down, for the interface associated with `ifIndex.1` on the host known as `giants`. The **-c** flag specifies the community for the host. The **-t 2** flag

specifies that the **snmpinfo** command will transmit the SNMP request to the SNMP agent 2 times before terminating if no response is received from the SNMP agent.

5. To dump a group of the MIB tree in verbose mode, enter a command similar to the following:

   ```
   snmpinfo -m dump -v interfaces
   ```

   In this example the `interfaces` group is dumped in verbose mode.

6. To dump the entire MIB tree, enter:

   ```
   snmpinfo -m dump
   ```

Files

/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle.

Related Information

The **mosy** command.

Understanding the Simple Network Management Protocol (SNMP), Using the Management Information Base (MIB) Database, and Understanding the Management Information Base (MIB) in *AIX 5L Version 5.3 Communications Programming Concepts*.

---

snmpmibd Daemon

Purpose

Starts the **snmpmibd** dpi2 sub-agent daemon as a background process.

Syntax

snmpmibd [**-f** *File*] [**-d** [*Level*]] [**-h** *HostName*] [**-c** *Community*]

Description

The **snmpmibd** command starts the **snmpmibd** dpi2 sub-agent. This command may only be issued by a user with root privileges or by a member of the system group.

The **snmpmibd** daemon complies with the standard Simple Network Management Protocol Distributed Protocl Interface Version 2.0 defined by RFC 1592. It acts as a dpi2 sub-agent to communicate with the dpi2 agent through dpiPortForTCP.0 (1.3.6.1.4.1.2.2.1.1.1.0) which is defined in RFC1592 section 3.1.

The Management Information Base(MIB) is defined by RFC 1155. The specific MIB variables **snmpmibd** is managing are defined by the following RFCs:

RFC 1213
: MIB-II

RFC 1229
: Extension to the Generic-Interface MIB

RFC 1231
: IEEE 802.5 Token Ring MIB

RFC 1398
: Ethernet-like Interface Types MIB

RFC 1512
: FDDI MIB

Note: The ″**system**″ and ″**snmp**″ groups defined in RFC1213 are not implemented by **snmpdmibd** daemon. Instead they are implemented by **snmpdv3** agent.

The **snmpmibd** daemon is normally executed during system startup when **/etc/rc.tcpip** shell script is called.

The **snmpmibd** daemon should be controlled using the System Resource Controller(SRC). Entering **snmpmibd** at the command line is not recommended.

Use the following SRC commands to manipulate the **snmpmibd** daemon:

startsrc
Starts a subsystem, group of subsystems, or a subserver.

stopsrc
Stops a subsystem, group of subsystems, or a subserver.

refresh
Causes a subsystem or group of subsystems to reread the appropriate configuration file.

lssrc Gets the status of a subsystem, group of subsystems, or a subserver. If the user issuing the long status form of the **lssrc** command is not the root user, no community name information is displayed.

Flags

-c *Community*
Use specified community name. If -c flag is not specified, the default community name is **public**.

-d [*Level*]
Specifies tracing/debug level. The levels are:

8	DPI level 1
16	DPI level 2
32	Internal level 1
64	Internal level 2
128	Internal level 3

Add the numbers for multiple trace levels.

If **-d** flag is specified and the *Level* is not specified, the default level is 56.

If **-d** flag is not specified, the default level is 0.

-f *File*
A non-default configuration file. If the -f flag is not specified, the default configuration file is **/etc/snmpmibd.conf**. See **/etc/snmpmibd.conf** file for information on this file format.

-h *HostName*
Send request to specified host. If **-h** flag is not specified, the default destination host is loopback (127.0.0.1).

Examples

1. To start the **snmpmibd** daemon, enter a command similar to the following:

   ```
   startsrc -s snmpmibd -a "-f /tmp/snmpmibd.conf"
   ```

   This command starts the **snmpmibd** daemon and reads the configuration file from **/tmp/snmpmibd.conf**.
2. To stop the **snmpmibd** daemon normally, enter:

```
stopsrc -s snmpmibd
```

This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).

3. To get long status from the **snmpmibd** daemon, enter:

```
lssrc -ls snmpmibd
```

If you are the root user, this long form of the status report lists the configuration parameters in **/etc/snmpmibd.conf**.

Files

/etc/snmpmibd.conf	Defines the configuration parameters for snmpmibd command.
/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent and manager should recognize and handle.

Related Information

The **hostmibd** command, **snmpdv3** daemon.

---

snmptrap Command

Purpose

Generate a notification (trap) to report an event to the SNMP manager with the specified message.

Syntax

snmptrap [**-a** *Host*] [**-h** *TargetHost*] [**-c** *Community*] [**-d**] **-m** *Message*

Description

Generate a notification (trap) to report an event to the SNMP manager with the specified message.

Flags

-a *Host*	Specifies to connect to the SNMP agent on the specified host. If the **-a** flag is not specified, the default host is the local host. *Host* can be an IP address or a host name.
-c *Community*	Specifies community name to use. This community must have been set in **/etc/snmpdv3.conf** for SNMP version 3 or in **/etc/snmpd.conf** for SNMP version 1 and have the read access privilege at least to the SNMP agent running on the specified host or local host. If the **-c** flag is not specified, the default community name is ″public″.
-d	Enables the debug facility
-h *TargetHost*	Specifies the target network manger host to which the trap message will be sent. It is different from **-a** flag. The **-a** flag specifies a host where the AIX SNMP agent (**snmp**) must be running and the SNMP agent forwards this trap to network mangers. However, the **-h** flag does not require the AIX SNMP agent to forward the trap message to network managers, and it sends the trap directly to the network manager. If there are no **-h** and **-a** flags, the trap will be sent to the AIX SNMP agent on the local host.
-m *Message*	Defines the message that the **snmptrap** command will send. *Message* specifies the information the trap will hold. This information is in the text format.The **-m** flag must be the last flag specified.

Exit Status

0 Trap information was sent out correctly.

1 This indicates something was wrong during the process.

Examples

1. To send a trap with the message 'hello world' to the SNMP agent running on the local host, enter the following:

   ```
   snmptrap -m hello world
   ```

   Note: The community, public, must have read access to the SNMP agent running on the local host. For details, please refer to SNMP configuration documentation.
2. To send a trap with the community name, community1, and the message 'hello world' to the SNMP agent running on a remote host blah, enter the following:

   ```
   snmptrap -c community1 -h blah -m hello world
   ```

   Note: The community 'community1' must have read access to the SNMP agent running on the host 'blah'. For details, please refer to the SNMP configuration documentation.
3. To send a trap to the network manager running on a Linux® platform and where the host name is `nehcyg`, type the following:

   ```
   snmptrap -h nehcyg -m hello world
   ```

Files

/etc/snmpdv3.conf	Contains the configuration file for the SNMP version 3 agent.
/etc/snmpd.conf	Contains the configuration file for the SNMP version 1 agent.

Related Information

The **snmpdv3** daemon, **snmpdv1** daemon.

The SNMP for network management article in *Networks and communication management*

---

snmpv3_ssw Command

Purpose

Switch the symbolic links among the non-encrypted **snmpdv3** agent, encrypted **snmpdv3** agent and **snmpdv1** agent.

Syntax

snmpv3_ssw [**-e** | **-n** | **-1**]

Description

Switch the symbolic links among the non-encrypted snmpdv3 agent, encrypted snmpdv3 agent and snmpdv1 agent, and then start the newly choosed SNMP agent. A user can choose which version of SNMP agent to run.

For example, the if current running SNMP agent is the encrypted **snmpdv3** agent, the actual SNMP agent executable which is running on the machine is ″**/usr/sbin/snmpdv3e**″. The symblic links on the machine are:

- /usr/sbin/snmpd --> /usr/sbin/snmpdv3e
- /usr/sbin/clsnmp --> /usr/sbin/clsnmpe

If a user chooses to switch to the non-encrypted snmpdv3 agent, after user runs the **/usr/sbin/snmpv3_ssw** command with the **-n** option, the acutal snmp agent which is running on the machine ″**/usr/sbin/snmpdv3ne**″. The symbolic links on the machine will be changed to:

- /usr/sbin/snmpd --> /usr/sbin/snmpdv3ne
- /usr/sbin/clsnmp --> /usr/sbin/clsnmpne

Flags

-e	Switch to the encrypted version of snmpdv3 agent.
-n	Switch to the non-encrypted version of snmpdv3 agent.
-1	Switch to the snmpdv1 agent.

Examples

1. To switch to the encrypted version of **snmpdv3** agent, enter:

```
/usr/sbin/snmp3_ssw -e
```

Related Information

The **clsnmp** command, **hostmibd** command, **snmpdv1** daemon, **snmpdv3** daemon.

The **/etc/clsnmp.conf** file format, **/etc/snmpd.conf** file format, **/etc/snmpdv3.conf** file format.

---

sno Command

Purpose

Provides a SNOBOL interpreter.

Syntax

sno [*File* ...]

Description

The **sno** command provides a SNOBOL compiler and interpreter, with some differences from standard SNOBOL. It reads the named files and the standard input and compiles all input through a statement containing the **end** label. The rest is available to the **syspit** pseudo-variable.

The **sno** command differs from standard SNOBOL in the following ways:

- There are no unanchored searches. To get the same effect, use lines similar to the following:

`a ** b`	Produces an unanchored search for *b*.
`a *x* b = x c`	Produces an unanchored assignment.

- There is no back referencing.

```
x = ″abc″
```

`a *x* x` Produces an unanchored search for **abc**.

- Function declaration is done at compile time by the use of the (non-unique) **define** label. Execution of a function call begins at the statement following the **define** label. Functions cannot be defined at run time, and the use of the name **define** is preempted. There is no provision for automatic variables other than parameters. For example:

  ```
  define f()
  define f(a, b, c)
  ```
- All labels except **define** (even **end**), must have a nonempty statement.
- Labels, functions, and variables must all have distinct names. In particular, the nonempty statement on **end** cannot merely name a label.
- If **start** is a label in the program, program execution begins there. If not, execution begins with the first executable statement. The **define** label is not an executable statement.
- There are no built-in functions.
- Parentheses for arithmetic are not needed. Normal precedence applies. Because of this, the arithmetic operators **** (backslash) and ***** (asterisk) must be set off by spaces.
- The right side of assignments must be nonempty.
- Either **'** (single quotation mark) or ″ (double quotation mark) can be used for literal quotation marks.
- The pseudo-variable **sysppt** is not available.

Examples

To run the file `test.s` through the **sno** command and direct the output into the file output, enter:

```
sno < test.s > output
```

Files

/usr/bin/sno	Contains the **sno** command.

Related Information

The **awk** command.

---

sodebug Command

Purpose

Sets or unsets the socket debug flag (**SO_DEBUG** socket option) and trace level on sockets.

Syntax

sodebug [**-h**] [**-l** [*level*]] [**-p** *pid* | **-s** *sockaddr* [**-t** *type*]]

Description

The **sodebug** command sets, unsets, or lists the socket debug flag and trace level on active sockets

If the socket debug flag (also known as the **SO_DEBUG** socket option) is set for a socket, the events on this socket can be traced using the **trace** command.

You can use the **-l** option to set the socket debug flag on sockets that already exist on a system. The **-l** option also sets the trace level for a given socket.

If the **sodebug** command is run without any options, the socket debug flag status and trace level for each active socket displays.

The **trace** and **trpt** commands collect information based on the trace level.

The following table describes the information collected based on the trace level for trace hook ID 25A (TCPDBG):

Table 1.

	min	normal	detail
tcp_debug data (td_time, td_act, td_ostate, td_tcb, family and td_req)		X	X
tcpip header		X	X
Address of tcpcb		X	X
All tcpcb fields			X
Address of socket		X	X
All socket fields			X

You can also set or unset the socket debug flag and the trace level as described below:

1. The following command enables the socket debug flag for all sockets that are subsequently created on the system:

   ```
   no -o sodebug=1
   ```

2. You can specify l**DEBUG**[**=***level*] in the wait/nowait field of a service in inetd.conf to turn on socket debugging for a specific service. You can set the trace level to **min**, **normal**, or **detail**. If no level is specified, the default level is **normal**.
3. You can set socket debugging on or off for all subsequent sockets created by a process using the **sodebug_env** parameter of the **no** command and specifying **export SODEBUG=***level* in a process environment. You can set the trace level to **min**, **normal**, or **detail**.

Flags

-h — Displays help for the **sodebug** command.

-l [*level*] — Specifies the trace level. Valid values for level are **none**, **min**, **normal**, and **detail**. If no level is specified, the default trace level is **normal**.

-p *pid* — Specifies the process ID of a process.

-s *sockaddr* — Specifies a socket by the socket address, the address of the socket's inpcb, or the address of the socket's tcpcb.

-t *type* — Specifies the type of address that is specified by the **-s** *sockaddr* option. Valid values are **socket**, **inpcb**, and **tcpcb**. The default value is **socket**.

Security

You must have root authority to run the **sodebug** command.

Examples

1. To list the debug flag and socket trace level for socket f100090002d0a800, type:

   ```
   sodebug -s f100090002d0a800
   ```

   The output is similar to the following example:

   ```
   socket address : f100090002d0a800 , sodebug flag : 0 , trace level : none(0)
   ```

2. To set the trace level to normal and set the debug flag to 1, type:

   ```
   sodebug -s f100090002d0a800 -l normal
   ```

   The output is similar to the following example:

```
Setting new values for trace level and debug flag
socket address : f100090002d0a800 , sodebug flag : 1 , trace level : normal(3)
```

Related Information

The **trace** daemon and **trpt** command.

---

soelim Command

Purpose

Processes **.so** requests in **nroff** command files.

Syntax

soelim [*File* ... | **-**]

Description

The **soelim** command reads specified files or standard input and performs inclusion specified by the **nroff** command and **troff** command requests of the form `.so filename` when the request appears at the beginning of input lines. Any combination of ASCII spaces and ASCII tab characters can follow the **.so** request and precede the file name. No characters should follow the file name.

The **soelim** command is useful because commands, such as the **tbl** command, do not normally perform file inclusions during processing.

When the **-** (minus sign) flag is specified, a file name corresponding to standard input is included.

Flag

- Indicates a file name corresponding to standard input.

Note: Inclusion can be suppressed by using a ' (single quotation mark) instead of a **.** (period), as follows:

Parameter

File Specifies files that the command performs inclusion on. The default is standard input.

```
'so /usr/share/lib/tmac/tmac.s
```

Example

Following is a sample usage of the **soelim** command:

```
soelim exum?.n | tbl | nroff -ms -Tlp | col -Tlp | pg
```

In this example, you use the **soelim** command to preprocess the file inclusion (**.so**) requests. The output is then passed to the **tbl** command. This makes it easier to place tables in separate files that can be included in forming a large document.

Related Information

The **colcrt** command, **nroff** command, **tbl** command, **troff** command.

sort Command

Purpose

Sorts files, merges files that are already sorted, and checks files to determine if they have been sorted.

Syntax

sort [**-A**] [**-b**] [**-c**] [**-d**] [**-f**] [**-i**] [**-m**] [**-n**] [**-r**] [**-u**] [**-o** *OutFile*] [**-t** *Character*] [**-T** *Directory*] [**-y** [*Kilobytes*]] [**-z** *RecordSize*] [[**+** [*FSkip*] [**.***CSkip*] [**b**] [**d**] [**f**] [**i**] [**n**] [**r**]] [**-** [*FSkip*] [**.***CSkip*] [**b**] [**d**] [**f**] [**i**] [**n**] [**r**]]] ... [**-k** *KeyDefinition*] ... [*File ...*]

Description

The **sort** command sorts lines in the files specified by the *File* parameter and writes the result to standard output. If the *File* parameter specifies more than one file, the **sort** command concatenates the files and sorts them as one file. A **-**(minus sign) in place of a file name specifies standard input. If you do not specify any file names, the command sorts standard input. An output file can be specified with the **-o** flag.

If no flags are specified, the **sort** command sorts entire lines of the input file based upon the collation order of the current locale.

Sort Keys

A sort key is a portion of an input line that is specified by a field number and a column number. Fields are parts of input lines that are separated by field separators. The default field separator is a sequence of one or more consecutive blank characters. However, these blank characters are considered to be a part of the following field for sorting purposes. You can specify the **-b** option to ignore these leading blank characters. A different field separator can be specified using the **-t** flag. The tab and the space characters are the blank characters in the C and English Language locales.

When using sort keys, the **sort** command first sorts all lines on the contents of the first sort key. Next, all the lines whose first sort keys are equal are sorted upon the contents of the second sort key, and so on. Sort keys are numbered according to the order they appear on the command line. If two lines sort equally on all sort keys, the entire lines are then compared based upon the collation order in the current locale.

When numbering columns within fields, the blank characters in a default field separator are counted as part of the following field. Leading blanks are not counted as part of the first field, and field separator characters specified by the **-t** flag are not counted as parts of fields. Leading blank characters can be ignored using the **-b** flag.

Sort keys can be defined using the following two methods:

- **-k** *KeyDefinition*
- *FSkip***.***CSkip* (obsolescent version).

Sort Key Definition Using the -k Flag

The **-k** *KeyDefinition* flag uses the following form:

-k [*FStart* [**.***CStart*]] [*Modifier*] [**,** [*FEnd* [**.***CEnd*]][*Modifier*]]

The sort key includes all characters beginning with the field specified by the *FStart* variable and the column specified by the *CStart* variable and ending with the field specified by the *FEnd* variable and the column specified by the *CEnd* variable. If *Fend* is not specified, the last character of the line is assumed. If *CEnd* is not specified the last character in the *FEnd* field is assumed. Any field or column number in the *KeyDefinition* variable may be omitted. The default values are:

FStart Beginning of the line

CStart	First column in the field
FEnd	End of the line
CEnd	Last column of the field

If there is any spaces between the fields, **sort** considers them as separate fields.

The value of the *Modifier* variable can be one or more of the letters **b**, **d**, **f**, **i**, **n**, or **r**. The modifiers apply only to the field definition they are attached to and have the same effect as the flag of the same letter. The modifier letter **b** applies only to the end of the field definition to which it is attached. For example:

```
-k 3.2b,3r
```

specifies a sort key beginning in the second nonblank column of the third field and extending to the end of the third field, with the sort on this key to be done in reverse collation order. If the *FStart* variable and the *CStart* variable fall beyond the end of the line or after the *FEnd* variable and the *CEnd* variable, then the sort key is ignored.

A sort key can also be specified in the following manner:

[**+**[*FSkip1*] [*.CSkip1*] [*Modifier*]] [**-**[*FSkip2*] [*.CSkip2*] [*Modifier*]]

The **+***FSkip1* variable specifies the number of fields skipped to reach the first field of the sort key and the **+***CSkip* variable specifies the number of columns skipped within that field to reach the first character in the sort key. The **-***FSkip* variable specifies the number of fields skipped to reach the first character *after* the sort key, and the **-***CSkip* variable specifies the number of columns to skip within that field. Any of the field and column skip counts may be omitted. The defaults are:

FSkip1	Beginning of the line
CSkip1	Zero
FSkip2	End of the line
CSkip2	Zero

The modifiers specified by the *Modifier* variable are the same as in the **-k** flag key sort definition.

The field and column numbers specified by **+***FSkip1***.***CSkip1* variables are generally one less than the field and column number of the sort key itself because these variables specify how many fields and columns to skip before reaching the sort key. For example:

```
+2.1b -3r
```

specifies a sort key beginning in the second nonblank column of the third field and extending to the end of the third field, with the sort on this key to be done in reverse collation order. The statement `+2.1b` specifies that two fields are skipped and then the leading blanks and one more column are skipped. If the **+***FSkip1***.***CSkip1* variables fall beyond the end of the line or after the **-***FSkip2***.***CSkip2* variables, then the sort key is ignored.

Note: The maximum number of fields on a line is 32.

Flags

Note: A **-b**, **-d**, **-f**, **-i**, **-n**, or **-r** flag that appears before any sort key definition applies to all sort keys. None of the **-b**, **-d**, **-f**, **-i**, **-n**, or **-r** flags may appear alone after a **-k** *KeyDefinition*; if they are attached to a *KeyDefinition* variable as a modifier, they apply only to the attached sort key. If one of these flags follows a **+***Fskip.Cskip* or **-***Fskip.Cskip* sort key definition, the flag only applies to that sort key.

-A Sorts on a byte-by-byte basis using ASCII collation order instead of collation in the current locale.

-b Ignores leading spaces and tabs to find the first or last column of a field.

-c Checks that input is sorted according to the ordering rules specified in the flags. A nonzero value is returned if the input file is not correctly sorted.

-d Sorts using dictionary order. Only letters, digits, and spaces are considered in comparisons.

-f Changes all lowercase letters to uppercase before comparison.

-i Ignores all nonprinting characters during comparisons.

-k *KeyDefinition* Specifies a sort key. The format of the *KeyDefinition* option is:

[*FStart* [**.***CStart*]] [*Modifier*] [**,** [*FEnd* [**.***CEnd*]][*Modifier*]]

The sort key includes all characters beginning with the field specified by the *FStart* variable and the column specified by the *CStart* variable and ending with the field specified by the *FEnd* variable and the column specified by the *CEnd* variable. The value of the *Modifier* variable can be **b**, **d**, **f**, **i**, **n**, or **r**. The modifiers are equivalent to the flags of the same letter.

-m Merges multiple input files only; the input are assumed to be already sorted.

-n Sorts numeric fields by arithmetic value. A numeric field may contain leading blanks, an optional minus sign, decimal digits, thousands-separator characters, and an optional radix character. Numeric sorting of a field containing any nonnumeric character gives unpredictable results.

-o *OutFile* Directs output to the file specified by the *OutFile* parameter instead of standard output. The value of the *OutFile* parameter can be the same as the *File* parameter.

-r Reverses the order of the specified sort.

-t *Character* Specifies *Character* as the single field separator character.

-u Suppresses all but one line in each set of lines that sort equally according to the sort keys and options.

-T *Directory* Places all temporary files that are created into the directory specified by the *Directory* parameter.

-y[*Kilobytes*] Starts the **sort** command using the number of kilobytes of main storage specified by the *Kilobytes* parameter and adds storage as needed. (If the value specified in the *Kilobytes* parameter is less than the minimum storage site or greater than the maximum, the minimum or maximum is used instead). If the **-y** flag is omitted, the **sort** command starts with the default storage size. The **-y0** flag starts with minimum storage, and the **-y** flag (with no *Kilobytes* value) starts with maximum storage. The amount of storage used by the **sort** command affects performance significantly. Sorting a small file in a large amount of storage is wasteful.

-z *RecordSize* Prevents abnormal termination if any of the lines being sorted are longer than the default buffer size. When the **-c** or **-m** flags are specified, the sorting phase is omitted and a system default buffer size is used. If sorted lines are longer than this size, sort terminates abnormally. The **-z** option specifies recording of the longest line in the sort phase so adequate buffers can be allocated in the merge phase. *RecordSize* must designate a value in bytes equal to or greater than the longest line to be merged.

Exit Status

This command returns the following exit values:

0 All input files were output successfully, or **-c** was specified and the input file was correctly sorted.

1 Under the **-c** option, the file was not ordered as specified, or if the **-c** and **-u** options were both specified, two input lines were found with equal keys.

>1 An error occurred.

Examples

1. To sort the `fruits` file with the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to `En_US`, enter:

   ```
   LANG=En_US sort fruits
   ```

   This command sequence displays the contents of the `fruits` file sorted in ascending lexicographic order. The characters in each column are compared one by one, including spaces, digits, and special characters. For instance, if the `fruits` file contains the text:

   ```
   banana
   orange
   Persimmon
   apple
   %%banana
   apple
   ORANGE
   ```

   the **sort** command displays:

   ```
   %%banana
   ORANGE
   Persimmon
   apple
   apple
   banana
   orange
   ```

   In the ASCII collating sequence, the `%` (percent sign) precedes uppercase letters, which precede lowercase letters. If your current locale specifies a character set other than ASCII, your results may be different.

2. To sort in dictionary order, enter:

   ```
   sort  -d fruits
   ```

   This command sequence sorts and displays the contents of the `fruits` file, comparing only letters, digits, and spaces. If the `fruits` file is the same as in example 1, then the **sort** command displays:

   ```
   ORANGE
   Persimmon
   apple
   apple
   %%banana
   banana
   orange
   ```

   The **-d** flag ignores the `%` (percent sign) character because it is not a letter, digit, or space, placing `%%banana` with `banana`.

3. To group lines that contain uppercase and special characters with similar lowercase lines, enter:

   ```
   sort  -d  -f fruits
   ```

   The **-d** flag ignores special characters and the **-f** flag ignores differences in case. With the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to `C`, the output for the `fruits` file becomes:

   ```
   apple
   apple
   %%banana
   banana
   ORANGE
   orange
   Persimmon
   ```

4. To sort, removing duplicate lines, enter:

```
sort  -d  -f  -u fruits
```

The **-u** flag tells the **sort** command to remove duplicate lines, making each line of the file unique. This command sequence displays:

```
apple
%%banana
ORANGE
Persimmon
```

Not only is the duplicate `apple` removed, but `banana` and `ORANGE` as well. These are removed because the **-d** flag ignores the `%%` special characters and the **-f** flag ignores differences in case.

5. To sort as in example 4, removing duplicate instances unless capitalized or punctuated differently, enter:

```
sort  -u +0  -d  -f +0 fruits
```

Entering the `+0 -d -f` does the same type of sort that is done with `-d -f` in example 3. Then the `+0` performs another comparison to distinguish lines that are not identical. This prevents the **-u** flag from removing them.

Given the `fruits` file shown in example 1, the added `+0` distinguishes `%%banana` from `banana` and `ORANGE` from `orange`. However, the two instances of `apple` are identical, so one of them is deleted.

```
apple
%%banana
banana
ORANGE
orange
Persimmon
```

6. To specify the character that separates fields, enter:

```
sort  -t: +1 vegetables
```

This command sequence sorts the `vegetables` file, comparing the text that follows the first colon on each line. The `+1` tells the **sort** command to ignore the first field and to compare from the start of the second field to the end of the line. The `-t:` flag tells the **sort** command that colons separate fields. If `vegetables` contains:

```
yams:104
turnips:8
potatoes:15
carrots:104
green beans:32
radishes:5
lettuce:15
```

Then, with the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to C, the **sort** command displays:

```
carrots:104
yams:104
lettuce:15
potatoes:15
green beans:32
radishes:5
turnips:8
```

Note that the numbers are not in numeric order. This happened when a lexicographic sort compares each character from left to right. In other words, `3` comes before `5`, so `32` comes before `5`.

7. To sort numbers, enter:

```
sort  -t: +1  -n vegetables
```

This command sequence sorts the `vegetables` file numerically on the second field. If the `vegetables` file is the same as in example 6, then the **sort** command displays:

```
radishes:5
turnips:8
lettuce:15
potatoes:15
green beans:32
carrots:104
yams:104
```

8. To sort more than one field, enter:

   ```
   sort  -t: +1 -2  -n +0 -1  -r vegetables
   ```

   OR

   ```
   sort  -t:  -k2,2 n  -k1,1 r vegetables
   ```

   This command sequence performs a numeric sort on the second field (`+1 -2 -n`). Within that ordering, it sorts the first field in reverse alphabetic order (`+0 -1 -r`). With the **LC_ALL**, **LC_COLLATE**, or **LANG** environment variable set to C, the output looks like this:

   ```
   radishes:5
   turnips:8
   potatoes:15
   lettuce:15
   green beans:32
   yams:104
   carrots:104
   ```

   The command sorts the lines in numeric order. When two lines have the same number, they appear in reverse alphabetic order.

9. To replace the original file with the sorted text, enter:

   ```
   sort  -o vegetables vegetables
   ```

   This command sequence stores the sorted output into the `vegetables` file (`-o vegetables`).

Files

/usr/bin/sort	Contains the **sort** command.
/var/tmp	Temporary space during the **sort** command processing.
/usr/tmp	Temporary space during the **sort** command processing, if file cannot be created in **/var/tmp**.
/tmp	Temporary space during the **sort** command processing, if file cannot be created in */var/tmp* or **/usr/tmp**.

Related Information

The **comm** command, **join** command, and **uniq** command.

Files, Input and output redirection in *Operating system and device management*.

National Language Support in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

sortbib Command

Purpose

Sorts a bibliographic database.

Syntax

sortbib [**-s***Keys*] [*Database* ...]

Description

The **sortbib** command sorts files of records containing **refer** command key letters by user-specified keys. The records can be separated by blank lines, or enclosed by the .[(period, left bracket) and the .] (period, right bracket) delimiters, but the two styles cannot be mixed together. The **sortbib** command reads through each database specified by the *Database* parameter and pulls out key fields, which are sorted separately. The sorted key fields contain the file pointer, byte offset, and length of corresponding records. These records are delivered using disk seeks and reads, so the **sortbib** command cannot be used in a pipeline to read standard input.

By default, the **sortbib** command alphabetizes by the first `%A` and `%D` fields, which contain the senior author and date.

The **sortbib** command sorts by the last word in the `%A` field, which is assumed to be the author's last name. A word in the final position, such as `jr.` or `ed.`, is ignored if the name preceding ends with a comma. Authors with two-word last names, or names with uncommon constructions, can be sorted correctly by using the **nroff** command convention **\0** in place of a space character. Specifying the `%Q` field is similar to the `%A` field, except sorting begins with the first, not the last, word.

> **Note:** Records with missing author fields should be sorted by title.

The **sortbib** command sorts by the last word of the `%D` line, which is usually the year. It ignores leading articles when sorting by titles in the `%T` or `%J` fields. The articles ignored are specific to the locale and specified in the locale-specific **refer** message catalog. Within this catalog, the articles are contained in a single message. Each article is separated by any number of ASCII space or tab characters. If a sort-significant field is absent from a record, the **sortbib** command places the record before other records containing that field.

No more than 16 databases can be sorted together at one time. Records longer than 4096 characters are truncated.

The *Database* parameter contains **refer** command key letters by user-specified keys that the **sortbib** command sorts through.

Flags

-s*Keys* Specifies field keys to sort on.

Examples

1. To sorts by author, title, and date:
   ```
   sortbib -sATD Database
   ```
2. To sort by author and date:
   ```
   sortbib -sA+D Database
   ```

Files

/tmp/Sbib*XXXXX*	Contains the temporary file.
/usr/bin/sort	Contains the **sort** command.

Related Information

The **addbib** command, **indxbib** command, **lookbib** command, **refer** command, **roffbib** command, **sort** command.

The **refer** message catalog in the *AIX 5L Version 5.3 National Language Support Guide and Reference*.

---

sortm Command

Purpose

Sorts messages.

Syntax

sortm [**+***Folder*] [*Messages*] [**-datefield** *Field*] [**-noverbose** | **-verbose**]

Description

The **sortm** command sorts messages according to their `Date:` field and renumbers them consecutively beginning with number one. Messages that are in the folder, but not specified to be sorted, are placed after the sorted messages. The **sortm** command displays a message if it cannot parse a date field.

To specify a field other than the `Date:` field, specify the **-datefield** flag. If you specify a folder, it becomes the current folder. The current message remains the current message for the specified folder, even if it moves during the sort.

Flags

-datefield *Field*	Specifies the header field to be used in the sort. The `Date:` field is the default.
+*Folder*	Specifies the folder with messages to be sorted. The default is the current folder.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.

Messages Specifies the messages to be sorted. Use the following references to specify messages:

Number
Number of the message.

Sequence
A group of messages specified by the user. Recognized values are:

all All messages in a folder. This is the default.

cur or . (period)
Current message.

first First message in a folder.

last Last message in a folder.

next Message following the current message.

prev Message preceding the current message.

-noverbose Prevents display of information during the sort. This flag is the default.

-verbose Displays information during the sort. This information allows you to monitor the steps involved.

Profile Entries

The following entries are found in the *UserMhDirectory*/**.mh_profile** file:

`Current-Folder:` Sets the default current folder.

`Path:` Specifies the *UserMhDirectory*.

Examples

1. To sort all the messages in the current folder according to the date, enter:

   ```
   sortm
   ```

2. To sort messages `5` through `10` in the `easter` folder according to the date, enter:

   ```
   sortm  +easter 5-10
   ```

Files

$HOME/.mh_profile Contains the MH user profile.

/usr/bin/sortm Contains the **sortm** command.

Related Information

The **folder** command.

The **.mh_alias** file format, **.mh_profile** file format.

Mail applications in *Networks and communication management*.

spell Command

Purpose

Finds English Language spelling errors.

Syntax

spell [**-b**] [**-i**] [**-l**] [**-v**] [**-x**] [**-d** *HashList*] [**-h** *HistoryList*] [**-s** *HashStop*] [**+** *WordList*] [*File* ...]

Description

The **spell** command reads words in the file indicated by the *File* variable and compares them to those in a spelling list. Words that cannot be matched in the spelling list or derived from words in the spelling list (by applying certain inflections, prefixes, and suffixes) are written to standard output. If no file name is specified, the **spell** command reads standard input.

The **spell** command ignores the same **troff**, **tbl**, and **eqn** codes as does the **deroff** command.

The coverage of the spelling list is uneven. You should create your own dictionary of special words used in your files. Your dictionary is a file containing a sorted list of words, one per line. To create your dictionary, use the **spellin** command.

Files containing an alternate spelling list, history list, and stop list can be specified by file name parameters following the **-d**, **-f**, and **-h** flags. Copies of all output can be accumulated in the history file.

Three programs help maintain and check the hash lists used by the **spell** command:

/usr/lbin/spell/hashmake	Reads a list of words from standard input and writes the corresponding 9-digit hash code to standard output.
/usr/bin/spellin *Number*	Reads the specified *Number* of hash codes from standard input and writes a compressed spelling list to standard output.
/usr/lbin/spell/hashcheck *SpellingList*	Reads a compressed *SpellingList*, recreates the 9-digit hash codes for all the words in it, and writes these codes to standard output.

The *File* parameter specifies the files that the **spell** command reads and compares them with the spelling list. If no file is specified, the command reads standard input.

Flags

-b	Checks British spelling. However, this flag does not provide a reasonable prototype for British spelling. The algorithms to derive a match against the spelling dictionary by applying certain inflections, prefixes, and suffixes are based on American English spelling.
-d *HashList*	Specifies the *HashList* file as the alternative spelling list. The default is **/usr/share/dict/hlist[ab]**.
-h *HistoryList*	Specifies the *HistoryList* file as the alternative history list, which is used to accumulate all output. The default is **/usr/lbin/spell/spellhist**. **Note:** The *HistoryList* file must be an existing file with read and write permissions.
-i	Suppresses processing of include files.

-l	Follows the chain of all include files (**.so** and **.nx** formatting commands). Without this flag, the **spell** command follows chains of all include files except for those beginning with **/usr/lib**.
-s *HashStop*	Specifies the *HashStop* file as the alternative stop list, which is used to filter out misspellings that would otherwise pass. The default is **/usr/share/dict/hstop**.
-v	Displays all words not in the spelling list and indicates possible derivations from the words.
-x	Displays every possible word stem with an = (equal sign).
+ *WordList*	Checks *WordList* for additional word spellings. *WordList* is the name of a file you provide that contains a sorted list of words, one per line. With this flag, you can specify a set of correctly spelled words (in addition to the **spell** command's own spelling list) for each job.

Exit Status

The following exit values are returned:

0	Indicates successful completion.
>0	Indicates an error occurred.

Examples

1. To check your spelling, enter:

   ```
   spell chap1 >mistakes
   ```

   This creates a file named `mistakes` containing all the words found in `chap1` that are not in the system spelling dictionary. Some of these may be correctly spelled words that the **spell** command does not recognize. Save the output of the **spell** command in a file because the word list may be long.
2. To check British spelling, enter:

   ```
   spell  -b chap1 >mistakes
   ```

   This checks `chap1` against the British dictionary and writes the questionable words in the `mistakes` file.
3. To see how the **spell** command derives words, enter:

   ```
   spell  -v chap1 >deriv
   ```

   This lists words not found literally in the dictionary but are derived from forms of dictionary words. The prefixes and suffixes used to form the derivations are indicated for each word. Words that are not in the dictionary at all are also listed.
4. To check your spelling against an additional word list, enter:

   ```
   spell  +newwords chap1
   ```

   This checks the spelling of words in `chap1` against the system dictionary and against `newwords`. The `newwords` file lists words in alphabetical order, one per line. You can create this file with a text editor, such as the ed editor, and alphabetize it with the **sort** command.

Files

/usr/share/dict/hlist[ab]	Contains hashed spelling lists, both American and British.
/usr/share/dict/hstop	Contains a hashed stop list.
/usr/lbin/spell/spellhist	Contains a history file.
/usr/lbin/spell/compress	Contains an executable shell program to compress the history file.

/usr/lbin/spell/hashmake	Creates hash codes from a spelling list.
/usr/bin/spellin *Number*	Creates spelling list from hash codes.
/usr/lbin/spell/hashcheck *SpellingList*	Creates hash codes from a compressed spelling list.
/usr/lbin/spell/spellinprg	Main program called by the **spellin** file.
/usr/lbin/spell/spellprog	Checks spelling.

Related Information

The **deroff** command, **eqn** command, **neqn** command, **sed** command, **sort** command, **spellin** command, **spellout** command, **tbl** command, **tee** command, and **troff** command.

---

spellin Command

Purpose

Creates a spelling list.

Syntax

spellin [*List* | *Number*]

Description

The **spellin** command creates a spelling list for use by the **spell** command. The parameter for the **spellin** command can be a file name or a number. The **spellin** command combines the words from the standard input and the already existing spelling list file and places a new spelling list on the standard output. If no list file is specified, a new list is created. If *Number* is specified, the **spellin** command reads the specified number of hash codes from standard input and writes a compressed spelling list.

Examples

To add the word `hookey` to the spelling list named `myhlist`, enter:

```
echo hookey | spellin /usr/share/dict/hlista > myhlist
```

Related Information

The **spell** command.

---

spellout Command

Purpose

Verifies that a word is not in the spelling list.

Syntax

spellout [**-d**] *List*

Description

The **spellout** command looks up each word from standard input and prints on standard output those that are missing from the hashed list file specified by the *List* parameter. The hashed list file is similar to the dictionary file used by the **spell** command.

Flags

-d Prints those words that are present in the hashed list file.

Examples

To verify that the word `hookey` is not on the default spelling list, enter:

```
echo hookey | spellout /usr/share/dict/hlista
```

In this example, the **spellout** command prints the word `hookey` on standard output if it is not in the hashed list file. With the **-d** flag, **spellout** prints the word `hookey` if it is found in the hash file.

Related Information

The **spell** command, **spellin** command.

---

split Command

Purpose

Splits a file into pieces.

Syntax

To Split a File Into Multiple Files Containing a Specified Number of Lines

split [**-l** *LineCount*] [**-a** *SuffixLength*] [*File* [*Prefix*]]

To Split a File Into Multiple Files Containing a Specified Number of Bytes

split -b *Number* [**k** | **m**] [**-a** *SuffixLength*] [*File* [*Prefix*]]

Description

The **split** command reads the specified file and writes it in 1000-line pieces to a set of output files. The name of the first output file is constructed by combining the specified prefix (*x* by default) with the *aa* suffix, the second by combining the prefix with the *ab* suffix, and so on lexicographically through *zz* (a maximum of 676 files). The number of letters in the suffix, and consequently the number of output name files, can be increased by using the **-a** flag.

You cannot specify a *Prefix* longer than **PATH_MAX** - 2 bytes (or **PATH_MAX** - *SuffixLength* bytes if the **-a** flag is specified). The **PATH_MAX** variable specifies the maximum path-name length for the system as defined in the **/usr/include/sys/limits.h** file.

If you do not specify an input file or if you specify a file name of - (minus sign), the **split** command reads standard input.

The **split** command can be used with any regular text or binary files. After a file has been split, it can be restored to its original form by using the **cat** command, and the file fragments will be listed in the appropriate order.

Flags

Note: The **-b** and **-l** flags are mutually exclusive.

-a *SuffixLength* Specifies the number of letters to use in forming the suffix portion of the output name files. The number of letters determines the number of possible output filename combinations. The default is two letters.

-b *Number* Splits the file into the number of bytes specified by the *Number* variable. Adding the *k* (kilobyte) or *m* (megabyte) multipliers to the end of the *Number* value causes the file to be split into *Number**1024 or *Number**1,048,576 byte pieces, respectively.

-l *LineCount* Specifies the number of lines in each output file. The default is 1000 lines.

Exit Status

This command returns the following exit values:

0 The command ran successfully.

>0 An error occurred.

Examples

1. To split a file into 1000-line segments, enter:

   ```
   split book
   ```

   This example splits `book` into 1000-line segments named `xaa`, `xab`, `xac`, and so forth.
2. To split a file into 50-line segments and specify the file-name prefix, enter:

   ```
   split -l 50 book sect
   ```

   This example splits `book` into 50-line segments named `sectaa`, `sectab`, `sectac`, and so forth.
3. To split a file into 2KB segments, enter:

   ```
   split -b 2k book
   ```

   This example splits the book into 2*1024-byte segments named `xaa`, `xab`, `xac`, and so forth.
4. To split a file into more than 676 segments, enter:

   ```
   split -l 5 -a 3 book sect
   ```

   This example splits a book into 5-line segments named `sectaaa`, `sectaab`, `sectaac`, and so forth, up to `sectzzz` (a maximum of 17,576 files).

Files

/usr/bin/split Contains the **split** command.

Related Information

The **cat** and **csplit** commands.

Files in *Operating system and device management*.

Input and output redirection in *Operating system and device management*.

---

splat Command

Purpose

Simple Performance Lock Analysis Tool (splat).
Provides kernel and **pthread** lock usage reports.

Syntax

splat -i *file* [**-n** *file*] [**-o** *file*] [**-d** [bfta]] [**-l** *address*] [**-c** *class*] [**-s** [acelmsS]] [**-C** *cpus*] [**-S** *count*] [**-t** *start*] [**-T** *stop*] [**-p**]

splat -h [*topic*]

splat -j

Description

splat (Simple Performance Lock Analysis Tool) is a software tool which post-processes AIX trace files to produce kernel simple and complex lock usage reports. It also produces **pthread** mutex read-write locks, and condition variables usage reports.

Flags

-i *inputfile* AIX trace file (REQUIRED).

-n *namefile* File containing output of **gensyms** command.

-o *outputfile* File to write reports to (DEFAULT: **stdout**).

-d *detail* Detail can be one of:
[**b**]asic: summary and lock detail (DEFAULT)
[**f**]unction: basic + function detail
[**t**]hread: basic + thread detail
[**a**]ll: basic + function + thread detail

-c *class* If the user supplies a decimal lock class index, **splat** will only report activity for locks in that class.

-l *address* If the user supplies a hexadecimal lock address, **splat** will only report activity for the lock at that address. **splat** will filter a trace file for lock hooks containing that lock address and produce a report solely for that lock.

-s *criteria* Sort the lock, function, and thread reports by the following criteria:

a	acquisitions
c	percent CPU hold time
e	percent elapsed hold time
l	lock address, function address, or thread ID
m	miss rate
s	spin count
S	percent CPU spin hold time (DEFAULT)
w	percent real wait time
W	average waitq depth

-C *cpus* Specify the number of CPUs present for this trace.

-S *count* The maximum number of entries in each report (DEFAULT: 10).

-t *starttime* Time offset in seconds from the beginning of the trace.

-T *stoptime* Time offset in seconds from the beginning of the trace to stop analyzing trace data. (DEFAULT: the end of the trace.)

-h [*topic*] Help on usage or a specific topic. Valid topics are:

- all
- overview
- input
- names
- reports
- sorting

-j Print a list of trace hooks used by splat.

-p Specifies the use of the PURR register to calculate CPU times.

Help

The following is a list of available help topics and a brief summary of each:

OVERVIEW	This text.
INPUT	AIX trace hooks required in order to acquire useful output from **splat**.
NAMES	What name utilities can be used to cause **splat** to map addresses to human-readable symbols.
REPORTS	A description of each report that splat can produce and the formulas used to calculate reported values.
SORTING	A list of all the available sorting options and how they are applied to **splat**'s output.

Splat Trace

Splat takes as primary input an AIX trace file which has been collected with the AIX trace command. Before analyzing a trace with **splat**, you will need to make sure that the trace is collected with an adequate set of hooks, including the following:

```
106 DISPATCH
10C DISPATCH IDLE PROCESS
10E RELOCK
112 LOCK
113 UNLOCK
134 HKWD_SYSC_EXECVE
139 HKWD_SYSC_FORK
419 CPU PREEMPT
465 HKWD_SYSC_CRTHREAD
46D WAIT LOCK
46E WAKEUP LOCK
606 HKWD_PTHREAD_COND
607 HKWD_PTHREAD_MUTEX
608 HKWD_PTHREAD_RWLOCK
609 HKWD_PTHREAD_GENERAL
```

Capturing these lock and unlock trace events can cause serious performance degradation due to the frequency that locks are used in a multiprocessor environment. Therefore, lock trace event reporting is normally disabled. In order to enable lock trace event reporting, the following steps must be taken before a trace can be collected which will include lock trace events that splat requires (KornShell syntax):

```
1. bosboot -ad /dev/hdisk0 -L

2. shutdown -Fr

3. (reboot the machine)

4. locktrace -S

5. mkdir temp.lib; cd temp.lib

6. ln -s /usr/ccs/lib/perf/libpthreads.a

7. export LIBPATH=$PWD:$LIBPATH
```

Steps 1 through 3 are optional. They enable the display of kernel lock class names instead of addresses. Please refer to **bosboot(1)** for more information on **bosboot** and its flags. Steps 5 through 7 are necessary for activating the user **pthread** lock instrumentation; the **temp.lib** subdirectory can be put anywhere. Steps 1 through 7 are necessary in order for the report to be complete.

Splat Names

Splat can take the output of **gensyms** as an optional input and use it to map lock and function addresses to human-readable symbols.

Lock classes and **offsets** can be used to identify a lock broadly, but not as specifically as the actual symbol.

Splat Reports

The report generated by **splat** consists of a report summary, a lock summary report section, and a list of lock detail reports, each of which may have an associated function detail and/or thread detail report.

```
Report Summary
^^^^^^^^^^^^^^
 The report summary consists of the following elements:

 - The trace command used to collect the trace.
 - The host that the trace was taken on.
 - The date that the trace was taken on.
 - The duration of the trace in seconds.
 - The estimated number of CPUs
 - The combined elapsed duration of the trace in seconds;
   ( the duration of the trace multiplied by the number of
     CPUs identified during the trace ).
 - Start time, which is the offset in seconds from the beginning of the
   trace that trace statistics begin to be gathered.
 - Stop time, which is the offset in seconds from the beginning of the
   trace that trace statistics stop being gathered.
 - Total number of acquisitions during the trace.
 - Acquisitions per second, which is computed by dividing
   the total number of lock acquisitions by the real-time
   duration of the trace.
 - % of Total Spin Time, this is the summation of all lock spin hold
   times, divided by the combined trace duration in seconds, divided by 100.
   The current goal is to have this value be less than 10% of the total
   trace duration.


 Lock Summary
 ^^^^^^^^^^^^

 The lock summary report has the following fields:

Lock           The name, lockclass or address of the lock.


Type              The type of the lock, identified by one of the following letters:
                      Q    A RunQ lock
                      S    A simple kernel lock
                      D    A disabled simple kernel lock
                      C    A complex kernel lock
                      M    A PThread mutex
                      V    A PThread condition-variable
                      L    A PThread read/write lock
Acquisitions     The number of successful lock attempts for this lock, minus
                  the number of times a thread was preempted while holding
                  this lock.

Spins            The number of unsuccessful lock attempts for this lock,
                  minus the number of times a thread was undispatched while
                  spinning.

Wait             The number of unsuccessful lock attempts that
 or               resulted in the attempting thread going to
Transform         sleep to wait for the lock to become available,
                  or allocating a krlock.
```

```
%Miss            Spins divided by Acquisitions plus Spins, multiplied by 100.

%Total           Acquisitions divided by the total number of all
                  lock acquisitions, multiplied by 100.

Locks/CSec       Acquisitions divided by the combined elapsed
                 duration in seconds.

Percent HoldTime
Real CPU          The percent of combined elapsed trace time that
                  threads held the lock in question while dispatched.
                  DISPATCHED_HOLDTIME_IN_SECONDS divided by combined
                  trace duration, multiplied by 100.

Real Elaps(ed)    The percent of combined elapsed trace time that
                  threads held the lock while dispatched or sleeping.
                  UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                  by combined trace duration, multiplied by 100.

Comb Spin         The percent of combined elapsed trace time that
                  threads spun while waiting to acquire this lock.
                  SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                  duration, multiplied by 100.
```

The lock summary report defaults to a list of ten locks, sorted in descending order by percent spin hold time (the tenth field). The length of the summary report can be adjusted using the **-S** switch. The sorted order of the summary report (and all other reports) can be set with the **-s** switch whose options are described in the SORTING help section, **splat -h** sorting.

```
Lock Detail
^^^^^^^^^^^

 The lock detail report consists of the following fields:

 LOCK              The address (in hexadecimal) of the lock.

 NAME              The symbol mapping for that address (if available)

 CLASS             The lockclass name (if available) and hexadecimal offset,
                   used to allocate this lock ( lock_alloc() kernel service ).

Parent Thread     Thread id of the parent thread. This field only exists for Mutex,
                   Read/Write lock and Conditional Variable report.

 creation time     Elapsed time in seconds after the first event recorded in trace,
                   if available. This field only exists for Mutex, Read/Write lock
                   and Conditional Variable report.
 deletion time     Elapsed time in seconds after the first event recorded in trace,
                   if available. Tthis field only exists for Mutex, Read/Write lock
                   and Conditional Variable report.

 Pid               Pid number associated to the lock (this field only exists for Mutex,
                   Read/Write lock and Conditional Variable report).

 Process Name     Process name associated to the lock (this field only exists for Mutex,
                  Read/Write lock and Conditional Variable report).

 Call-Chain       Stack of called methods (if possible to have them, this field only
                  exists for Mutex, Read/Write lock and Conditional Variable report).


 Acquisitions      The number of successful lock attempts for this lock.
                   This field is named Passes for the conditional variable lock report.

 Miss Rate         The number of unsuccessful lock attempts divided by
```

```
                Acquisitions plus unsuccessful lock attempts, multiplied
                by 100.

 Spin Count     The number of unsuccessful lock attempts.

 Wait Count     The number of unsuccessful lock attempts that resulted in
                the attempting thread going to sleep to wait for the lock
                to become available.

 Transform Count  The number of krlock allocated and deallocated by the simple lock.

 Busy Count     The number of simple_lock_try() calls that returned busy.

 Seconds Held
 CPU            The total time in seconds that this lock was held by
                dispatched threads.

 Elapsed        The total time in seconds that this lock was held by
                both dispatched and undispatched threads.

NOTE: neither of these two values should exceed the
      total real elapsed duration of the trace.

 Percent Held
 Real CPU       The percent of combined elapsed trace time that
                threads held the lock in question while dispatched.
                DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                duration, multiplied by 100.

 Real Elaps(ed) The percent of combined elapsed trace time that
                threads held the lock while dispatched or sleeping.
                UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                by trace duration, multiplied by 100.

 Comb Spin      The percent of combined elapsed trace time that
                threads spun while waiting to acquire this lock.
                SPIN_HOLDTIME_IN_SECONDS divided by trace duration,
                multiplied by 100.

 Wait           The percentage of combined elapsed trace time that
                threads unsuccessfully tried to acquire this lock.

 SpinQ          Splat keeps track of the minimum, maximum and average
                depth of the spin queue (the threads spinning, waiting
                for a lock to become available).

 WaitQ          As with the spin queue, splat also tracks the minimum,
                maximum and average depth of the queue of threads waited
                waiting for a lock to become available).

PROD            The associated krlocks prod calls count.

CONFER SELF     The confer to self calls count for the simple lock and the
                associated krlocks.

CONFER TARGET   The confer to target calls count for the simple lock and the
                associated krlocks. w/ preemption reports the successfull calls
                count, resulting in a preemption.

CONFER ALL      The confer to all calls count for the simple lock and the
                associated krlocks. w/ preemption reports the successfull calls
                count, resulting in a preemption.

HANDOFF         The associated krlocks handoff calls count.

Lock Activity w/Interrupts Enabled (mSecs)
```

This section of the lock detail report are dumps of the raw data that splat collects for each lock, times expressed in milliseconds. The five states: LOCK, SPIN, WAIT, UNDISP(atched) and PREEMPT are the five basic states of **splat**'s enabled **simple_lock** finite state machine. The count for each state is the number of times a thread's actions resulted in a transition into that state. The durations in milliseconds show the minimum, maximum, average and total amounts of time that a lock request spent in that state.

```
   LOCK:    this state represents a thread successfully acquiring a lock.

   SPIN:    this state represents a thread unsuccessfully trying to acquire
            a lock.

   WAIT:    this state represents a spinning thread (in SPIN) going to sleep
            (voluntarily) after exceeding the thread's spin threshold.

   UNDISP:  this state represents a spinning thread (in SPIN) becoming
            undispatched (involuntarily) before exceeding the thread's
            spin threshold.

   PREEMPT: this state represents when a thread holding a lock is
            undispatched.


Lock Activity w/Interrupts Disabled (mSecs)
```

This section of the lock detail report are dumps of the raw data that splat collects for each lock, times expressed in milliseconds. The six states: LOCK, SPIN, LOCK with KRLOCK, KRLOCK LOCK, KRLOCK SPIN and TRANSFORM are the six basic states of **splat**'s disabled **simple_lock** finite state machine. The count for each state is the number of times a thread's actions resulted in a transition into that state. The durations in milliseconds show the minimum, maximum, average and total amounts of time that a lock request spent in that state.

```
LOCK:         This state represents a thread successfully acquiring a lock.

SPIN:         This state represents a thread unsuccessfully trying to acquire
              a lock.

LOCK with     The thread has successfully acquired the lock, while holding
 KRLOCK:      the associated krlock, and is currently executing.

KRLOCK LOCK: The thread has successfully acquired the associated krlock,
              and is currently executing.

KRLOCK SPIN: The thread is executing and unsuccessfully attempting to acquire
              the associated krlock.

TRANSFORM:    The thread has successfully allocated a krlock it associates to,
              and is executing.


Function Detail
^^^^^^^^^^^^^^^

 The function detail report consists of the following fields:

 Function Name   The name or return address of the function which
                 used the lock.

 Acquisitions    The number of successful lock attempts for this lock.
                 For complex lock and read/write lock there is a
                 distinction between acquisition for writing
                 (Acquisition Write) and for reading
                 (Acquisition Read).


 Miss Rate      The number of unsuccessful lock attempts divided by
                Acquisitions, multiplied by 100.
```

```
 Spin Count     The number of unsuccessful lock attempts.
                For complex lock and read/write lock there is a
                distinction between spin count for writing
                (Spin Count Write) and for reading
                (Spin Count Read).

 Wait Count     The number of unsuccessful lock attempts that resulted in
                the attempting thread going to sleep to wait for the lock
                to become available.
                For complex lock and read/write lock there is a
                distinction between wait count for writing
                (Wait Count Write) and for reading
                (Wait Count Read).

Transform Count The number of times that a simple lock has allocated a krlock,
                while the thread was trying to acquire the simple lock.

 Busy Count      The number of simple_lock_try() calls that returned busy.

 Percent Held of Total Time
 CPU             The percent of combined elapsed trace time that
                 threads held the lock in question while dispatched.
                 DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                 duration, multiplied by 100.

 Elaps(ed)       The percent of combined elapsed trace time that
                 threads held the lock while dispatched or sleeping.
                 UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                 by trace duration, multiplied by 100.

 Spin            The percent of combined elapsed trace time that
                 threads spun while waiting to acquire this lock.
                 SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                 duration, multiplied by 100.

 Wait            The percentage of combined elapsed trace time that
                 threads unsuccessfully tried to acquire this lock.

 Return Address  The calling function's return address in hexadecimal.

 Start Address   The start address of the calling function in hexadecimal.

 Offset          The offset from the function start address in hexadecimal.


Thread Detail
^^^^^^^^^^^^^

The thread detail report consists of the following fields:

 ThreadID        Thread identifier.

 Acquisitions    The number of successful lock attempts for this lock.

 Miss Rate       The number of unsuccessful lock attempts divided by
                 Acquisitions, multiplied by 100.

 Spin Count      The number of unsuccessful lock attempts.

 Wait Count      The number of unsuccessful lock attempts that resulted in
                 the attempting thread going to sleep to wait for the lock
                 to become available.

Transform Count The number of times that a simple lock has allocated a krlock,
                while the thread was trying to acquire the simple lock.
```

```
Busy Count      The number of simple_lock_try() calls that returned busy.

 Percent Held of Total Time
 CPU             The percent of combined elapsed trace time that
                 threads held the lock in question while dispatched.
                 DISPATCHED_HOLDTIME_IN_SECONDS divided by trace
                 duration, multiplied by 100.

 Elaps(ed)       The percent of combined elapsed trace time that
                 threads held the lock while dispatched or sleeping.
                 UNDISPATCHED_AND_DISPATCHED_HOLDTIME_IN_SECONDS divided
                 by trace duration, multiplied by 100.

 Spin            The percent of combined elapsed trace time that
                 threads spun while waiting to acquire this lock.
                 SPIN_HOLDTIME_IN_SECONDS divided by combined trace
                 duration, multiplied by 100.

 Wait            The percent of combined elapsed trace time that
                 threads unsuccessfully tried to acquire this lock.

 ProcessID       Process identifier (only for SIMPLE and COMPLEX Lock report).

 Process Name    Name of the process (only for SIMPLE and COMPLEX Lock report).
```

Splat Sorting

splat allows the user to specify which criteria is used to sort the summary and lock detail reports using the **-s** option. The default sorting criteria is to sort by percent spin hold time, which is the ratio of time that threads spent spinning for a lock compared to the combined duration of the trace. Using **-s**, the sort criteria can be changed to the following:

a Acquisitions; the number times a thread successfully acquired a lock.

c Percent CPU hold time; the ratio of CPU hold time with the combined trace duration.

e Percent Elapsed hold time; the ratio of elapsed hold time with the combined trace duration.

l location; the address of the lock or function, or the ID of a thread.

m Miss rate; the ratio missed lock attempts with the number of acquisitions.

s Spin count; the number of unsuccessful lock attempts that result in a thread spinning waiting for the lock.

S Percent CPU spin hold time (default).

w Percent elapsed wait time; the percent of the total time that a nonzero number of threads waited on the lock.

W Average **waitq** depth; the average number of threads waiting on the lock, equivalent to the average time each waiting thread spends in this state.

splat will use the specified criteria to sort the lock reports in descending order.

Restrictions

Other types of locks, such as VMM, XMAP, and certain Java-specific locks are not analyzed.

Files

/etc/bin/splat Simple Performance Lock Analysis Tool (**splat**). Provides kernel and pthread lock usage reports.

Related Information

The **simple_lock(3)**, **simple_unlock(3)**, **disable_lock(3)**, **unlock_enable(3)**, **trace(1)**, **trcrpt(1)**, **trcfmt(5)**, **gensyms(1)** and **bosboot(1)** daemons.

splitvg Command

Purpose

Splits a single mirror copy of a fully mirrored volume group.

Syntax

splitvg [**-y** *SnapVGname*] [**-c** Copy] [**-f**] [**-i**] ***VGname***

Description

The **splitvg** command splits a single mirror copy of a fully mirrored volume group into a snapshot volume group. The orginal volume group ***VGname*** will stop using the disks that are now part of the snapshot volume group ***SnapVGname***. Both volume groups will keep track of the writes within the volume group so that when the snapshot volume group is rejoined with the original volume group consistent data is maintained across the rejoined mirrors copies.

Notes:

1. To split a volume group, all logical volumes in the volume group must have the target mirror copy and the mirror must exist on a disk or set of disks. Only the target mirror copy must exist on the target disk or disks.
2. The **splitvg** command will fail if any of the disks to be split are not active within the orginal volume group.
3. In the unlikely event of a system crash or loss of quorum while running this command, the **joinvg** command must be run to rejoin the disks back to the orginal volume group.
4. There is no concurrent or enhanced concurrent mode support for creating snapshot volume groups.
5. New logical volumes and file system mount points will be created in the snapshot volume group.
6. The **splitvg** command is not supported for the rootvg.
7. The **splitvg** command is not supported for a volume group that has an active paging space.
8. When the **splitvg** command targets a concurrent-capable volume group which is varied on in non-concurrent mode, the new volume group that is created will not be varied on when the **splitvg** command completes. The new volume group must be varied on manually.

Flags

-y *SnapVGname*	Allows the volume group name to be specified rather than having the name generated automatically. Volume group names must be unique across the system and can range from 1 to 15 characters. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration database for other devices. The new volume group name is sent to standard output.
-c *Copy*	Which mirror to split. Valid values are 1, 2, or 3. The default is the second copy.
-f	Will force the split even if the mirror copy specified to create the snashot volume group has stale partitions.
-i	Will split the mirror copy of a volume group into a new volume group that can not be rejoined into the orginal.

Security

Access Control: You must have `root` authority to run this command.

Examples

1. To split a volume group, type:

   ```
   splitvg testvg
   ```

The second mirror copy of the volume group **testvg** is split into new volume group with an automatically generated name, which will be displayed.

2. To split first mirror copy of the volume group with the name **snapvg**, type:

```
splitvg -y snapvg -c 1 testvg
```

Files

/usr/sbin	Directory where the **splitvg** command resides.

Related Information

The **joinvg** and **recreatevg** commands.

---

splitlvcopy Command

Purpose

Splits copies from one logical volume and creates a new logical volume from them.

Syntax

splitlvcopy [**-f**] [**-y** *NewLogicalVolumeName*] [**-Y** *Prefix*] *LogicalVolume Copies* [*PhysicalVolume* ...]

Description

Notes:

1. To use this command, you must either have `root` user authority or be a member of the system group.
2. The **splitlvcopy** command is not allowed on a snapshot volume group or a volume group that has a snapshot volume group.

Attention: Although the **splitlvcopy** command can split logical volumes that are open, including logical volumes containing mounted filesystems, this is not recommended. You may lose consistency between *LogicalVolume* and *NewLogicalVolume* if the logical volume is accessed by multiple processes simultaneously. When splitting an open logical volume, you implicitly accept the risk of potential data loss and data corruption associated with this action. To avoid the potential corruption window, close logical volumes before splitting and unmount filesystems before splitting.

The **splitlvcopy** command removes copies from each logical partition in *LogicalVolume* and uses them to create *NewLogicalVolume*. The *Copies* parameter determines the maximum number of physical partitions that remain in *LogicalVolume* after the split. Therefore, if *LogicalVolume* has 3 copies before the split, and the *Copies* parameter is 2, *LogicalVolume* will have 2 copies after the split and *NewLogicalVolume* will have 1 copy. You can not split a logical volume so that the total number of copies in *LogicalVolume* and *NewLogicalVolume* after the split is greater than the number of copies in *LogicalVolume* before the split.

The *NewLogicalVolume* will have all the same logical volume characteristics as *LogicalVolume*. If *LogicalVolume* does not have a logical volume control block the command will succeed with a warning message and creates *NewLogicalVolume* without a logical volume control block.

There are additional considerations to take when splitting a logical volume containing a filesystem. After the split there will be two logical volumes but there will only be one entry in the **/etc/filesystems** file which refers to *LogicalVolume*. To access *NewLogicalVolume* as a filesystem you must create an additional entry in **/etc/filesystems** with a different mount point which refers to *NewLogicalVolume*. If the mount point does not already exist, you have to create it before the new filesystem can be mounted. In addition, if *NewLogicalVolume* was created while *LogicalVolume* was open, you have to run the command

```
fsck /dev/NewLogicalVolume
```

before the new filesystem can be mounted.

You can not use the System Management Interface Tool (SMIT) to run this command. Message catalogs are not supported for this command and therefore the error messages are provided in English only with no message catalog numbers. Documentation for splitlvcopy consists of this man page.

Flags

-f	Specifies to split open logical volumes without requesting confirmation. By default, **splitlvcopy** requests confirmation before splitting an open logical volume. This includes open raw logical volumes and logical volumes containing mounted filesystems.
-y *NewLogicalVolumeName*	Specifies the name of the new logical volume to move copies to from *LogicalVolume*.
-Y *Prefix*	Specifies the *Prefix* to use instead of the prefix in a system-generated name for the new logical volume. The prefix must be less than or equal to 13 characters. A name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices, nor be a name already used by another device.

Parameters

Copies	Specifies the maximum number of physical partitions that remain in LogicalVolume after the split.
LogicalVolume	Specifies the logical volume name or logical volume ID to split.
PhysicalVolume	Specifies the physical volume name or the physical volume ID to remove copies from.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command or be a member of the system group.

Auditing Events: N/A

Examples

To split one copy of each logical partition belonging to logical volume named **oldlv** which currenttly has 3 copies of each logical partition, and create the logical volume **newlv**, enter:

```
splitlvcopy -y newlv oldlv 2
```

Each logical partition in the logical volume **oldlv** now has two physical partitions. Each logical partition in the logical volume **newlv** now has one physical partition.

Files

/etc/splitlvcopy	Contains the **splitlvcopy** command.
/tmp	Contains the temporary files created while the **splitlvcopy** command is running.

Related Information

Commands: **rmlvcopy** and **mklv**.

---

splp Command

Purpose

Changes or displays printer driver settings.

Syntax

splp [**-b** *Option*] [**-B** *Number*] [**-c** *Option*] [**-C** *Option*] [**-e** *Option*] [**-f** *Option*] [**-F!**] [**-i** *Number*] [**-l** *Number*] [**-n** *Option*] [**-N** *Option*] [**-p** *Option*] [**-P** *Option*] [**-r** *Option*] [**-s** *Number*] [**-S** *Option*] [**-t** *Option*] [**-T** *Number*] [**-w** *Number*] [**-W** *Option*] [*DevicePath*]

Description

The **splp** command changes or displays settings for a printer device driver. The default device path is **/dev/lp0**; all flags are optional. If the device path does not begin with a **/** (backslash) character, the **/dev** directory is assumed. Also, if no flags are specified, the **splp** command reports the current settings for the specified device path. To change the current settings, specify the appropriate flags. No other processing is done, and there is no other output.

The changes that the **splp** command makes remain in effect until the next time you restart the system or rerun the **splp** command. The **splp** command can be run from the **/etc/inittab** command file to configure your printer each time you start up the system.

> **Note:** The **splp** command settings for the **-b**, **-c**, **-C**, **-f**, **-i**, **-l**, **-n**, **-p**, **-r**, **-t**, **-w**, and **-W** flags apply only when data is sent directly to the printer device (for example, redirecting the output of the **cat** command directly to the specifies device path). When files are queued for printing with the **enq**, **qprt**, **lp**, or **lpr** commands, the settings for these flags are ignored and are not changed.

Flags

-b *Option* Specifies whether backspaces are sent to the printer:

- **+** Specifies backspaces be sent to the printer.
- **!** Specifies backspaces be discarded.

-B *Number* Sets the speed to the specified number of bits per second. Values for the *Number* variable are 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19,200, and 38,400.

-c *Option* Specifies whether carriage returns are sent to the printer:

- **+** Sends carriage returns to the printer.
- **!** Translates carriage returns to line feeds.

-C *Option* Specifies whether all lowercase characters are converted to uppercase characters:

- **+** Converts lowercase characters to uppercase characters.
- **!** Does not convert lowercase characters to uppercase characters.

-e *Option* Specifies the processing to be performed when an error is detected:

- **+** Returns an error.
- **!** Waits until error clears.

-f *Option* Specifies whether the printer is sent form feeds or simulates a form feed with line feeds or carriage returns:

+ Sends form feeds to the printer.

! Simulates a form feed with line feeds or carriage returns.

-F! Resets font status indicators for an 3812 Page Printer or an 3816 Page Printer. This flag causes fonts to be reloaded from the printer's font diskette into the printer's memory by the next spooled print job. This flag should be specified if the printer has been turned off and then turned back on, or if the fonts in the printer's memory have become corrupted.

-i *Number* Indents the specified number of columns, where the value of the *Number* variable is an integer.

-l *Number* Prints the specified number of lines per page, where the value of the *Number* variable is an integer.

-n *Option* Specifies whether the printer is sent line feeds or translates line feeds to carriage returns:

+ Sends line feeds to the printer.

! Translates line feeds to carriage returns.

-N *Option* Specifies whether parity generation and detection is enabled:

+ Enables parity generation and detection.

! Disables parity generation and detection.

-p *Option* Specifies whether the system sends all characters to the printer unmodified or translates characters according to the settings for the **-b**, **-c**, **-C**, **-f**, **-i**, **-l**, **-n**, **-r**, **-t**, **-w**, and **-W** flags:

+ Sends all characters to the printer unmodified, overriding other settings.

! Translates characters according to the settings.

-P *Option* Specifies the parity:

+ Specifies odd parity.

! Specifies even parity.

-r *Option* Specifies whether carriage returns are added after line feeds:

+ Sends a carriage return after a line feed.

! Does not send a carriage return after a line feed.

-s *Number* Selects character size where the *Number* variable is the number of bits. Values for the *Number* variable can be 5, 6, 7, or 8. See the **termio.h** special file for additional information on character size.

-S *Option* Specifies the number of stop bits per character:

+ 2 stop bits per character.

! 1 stop bit per character.

-t *Option* Specifies whether tabs are to be expanded:

+ Does not expand tabs.

! Expands tabs on 8 position boundaries.

-T *Number* Sets the time-out period to the number of seconds specified by the *Number* variable. The value of the *Number* variable must be an integer.

-w *Number* Prints the number of columns specified by the *Number* variable. The value of the *Number* variable must be an integer.

-W *Option* Specifies whether to wrap characters beyond the specified width to the next line and print **...** (3 dots) after the new-line character:

+ Wraps characters beyond the specified width to the next line and prints **...** (3 dots) after the new-line character.

! Truncates characters beyond the specified width.

Examples

1. To display the current printer settings for the **/dev/lp0** printer, enter:

```
splp
```

2. To change the printer settings, enter:

```
splp   -w 80  -W +  -C +
```

   This changes the settings of the **/dev/lp0** printer for 80-column paper (the **-w 80** flag). It also wraps each line that is more than 80 columns wide onto a second line (the **-W+** flag), and prints all alphabetic characters in uppercase (the **-C+** flag).

Files

/dev/lp*	Contains the printer attribute file.
/etc/inittab	Contains the printer configuration command file.

Related Information

The **cat** command, **enq** command, **lp** command, **lpr** command, **qprt** command.

The **termio.h** file.

Printer Administration in the *Printers and printing*.

Adding a Printer Using the Printer Colon File in the *Printers and printing*.

Virtual Printer Definitions and Attributes in the *Printers and printing*.

---

spost Command

Purpose

Routes a message.

Syntax

spost [**-noalias** | **-alias** *File ...*] [**-format** | **-noformat**] [**-filter** *File* | **-nofilter**] [**-width** *Number*] [**-watch** | **-nowatch**] [**-remove** | **-noremove**] [**-backup** | **-nobackup**] [**-verbose** | **-noverbose**]*File*

Description

The **spost** command routes messages to the correct destinations. The **spost** command is not started by the user. The **spost** command is called by other programs only.

The **spost** command searches all components of a message that specify a recipient's address and parses each address to check for proper format. The **spost** command then puts addresses in the standard format and starts the **sendmail** command. The **spost** command performs a function similar to the **post** command, but it does less address formatting than the **post** command.

The **spost** command is the default (over the **post** command). Change the default by setting the **postproc** variable in your **.mh_profile**. For example:

```
postproc: /usr/lib/mh/post
```

The *File* parameter is the name of the file to be posted.

Flags

-alias *File*	Searches the specified mail alias file for addresses. You can repeat this flag to specify multiple mail alias files. The **spost** command automatically searches the **/etc/mh/MailAliases** file.
-backup	Renames the message file by placing a , (comma) before the file name after the **spost** command successfully posts the message.
-filter *File*	Uses the header components in the specified file to copy messages sent to the `Bcc:` field recipients.
-format	Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For Message Handler (MH), the name of this flag must be fully spelled out.
-noalias	Does not use any alias files for delivering the message.
-nobackup	Does not rename the message after posting the file. This flag is the default.
-nofilter	Strips the `Bcc:` field header from the message and sends it to recipients specified in the `Bcc:` component. This flag is the default.
-noformat	Does not alter the format of the recipient addresses.
-noremove	Does not remove the temporary message file after posting the message.
-noverbose	Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
-nowatch	Does not display information during delivery by the **sendmail** command. This flag is the default.
-remove	Removes the temporary message file after the message has been successfully posted. This flag is the default.
-verbose	Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved.
-watch	Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved.
-width *Number*	Sets the width of components that contain addresses. The default is 72 columns.

Files

$HOME/.mh_profile	Contains the Message Handler (MH) user profile.
/tmp/pst*Number*	Contains the temporary message file.
/etc/mh/MailAliases	Contains the default mail aliases.
/usr/lib/mh/.mh_profile	Contains the Message Handler (MH) user profile.

Related Information

The **ali** command, **conflict** command, **mhmail** command, **post** command, **send** command, **sendmail** command, and **whom** command.

The **.mh_alias** file format.

Mail applications in *Networks and communication management*.

---

spray Command

Purpose

Sends a specified number of packets to a host and reports performance statistics.

Syntax

/usr/sbin/spray *Host* [**-c** *Count*] [**-d** *Delay*] [**-i**] [**-l** *Length*]

Description

The **spray** command uses the Remote Procedure Call (RPC) protocol to send a one-way stream of packets to the host you specify. This command reports how many packets were received and at what transfer rate. The *Host* parameter can be either a name or an Internet address. The host only responds if the **sprayd** daemon is running.

Note: The **spray** command does not support IPv6.

See the **rpc.sprayd** daemon documentation for factors that affect **spray** command performance.

Flags

-c *Count* Specifies the number of packets to send. The default value is the number of packets required to make the total stream size 100,000 bytes.

-d *Delay* Specifies the time, in microseconds, the system pauses between sending each packet. The default is 0.

-i Uses the Internet Control Message Protocol (ICMP) echo packets rather than the RPC protocol. Since ICMP echoes automatically, it creates a two-way stream. You must be root user to use this option.

-l *Length* Specifies the number of bytes in the packet that holds the RPC call message. The default value of the *Length* parameter is 86 bytes, the size of the RPC and UDP headers.

The data in the packet is encoded using eXternal Data Representation (XDR). Since XDR deals only with 32-bit quantities, the **spray** command rounds smaller values up to the nearest possible value.

When the *Length* parameter is greater than 1500 for Ethernet or 1568 for token-ring, the RPC call can no longer fit into one Ethernet packet. Therefore, the *Length* field no longer has a simple correspondence to Ethernet packet size.

Examples

1. When sending a **spray** command to a workstation, specify the number of packets to send and the length of time the system will wait between sending each packet as follows:

   ```
   /usr/sbin/spray zorro  -c 1200 -d 2
   ```

   In this example, the `spray` command sends `1200` packets at intervals of `2` microseconds to the workstation named `zorro`.
2. To change the number of bytes in the packets you send, enter:

   ```
   /usr/sbin/spray zorro  -l 1350
   ```

   In this example, the `spray` command sends `1350`-byte packets to the workstation named `zorro`.
3. To send echo packets using the ICMP protocol instead of the RPC protocol, enter:

   ```
   /usr/sbin/spray zorro  -i
   ```

   In this example, the `spray` command sends echo packets to the workstation named `zorro`.

Related Information

The **sprayd** daemon.

List of NFS commands.

Network File System (NFS) in *Networks and communication management*.

NFS troubleshooting in *Networks and communication management*.

---

sprayd Daemon

Purpose

Receives packets sent by the **spray** command.

Syntax

/usr/lib/netsvc/spray/rpc.sprayd

Description

The **rpc.sprayd** daemon is a server that records the packets sent by the **spray** command. The **rpc.sprayd** daemon is normally started by the **inetd** daemon.

UDP Performance

User Datagram Protocol (UDP) performance with the **spray** command and the **rpc.sprayd** daemon can be affected by the following factors:

- How memory buffers (mbufs) are tuned for system configuration.
- The incoming burst rate (that is, interframe gap) of UDP packets for the **spray** command.
- Other system activity. Since the **rpc.sprayd** daemon runs as a normal user process, other activity (such as the **init** process, or the **syncd** daemon) can affect the operation of the **rpc.sprayd** daemon.
- Priority of the **rpc.sprayd** daemon process. The **rpc.sprayd** daemon has a floating process priority that is calculated dynamically.
- The size of the receive socket buffer used by the **rpc.sprayd** daemon. Because various implementations use different socket buffer sizes, measuring UDP performance with the **spray** command and the **rpc.sprayd** daemon is difficult and inconclusive.

Files

/etc/inetd.conf	TCP/IP configuration file that starts RPC daemons and other TCP/IP daemons.

Related Information

The **spray** command.

The **inetd** daemon.

List of NFS commands.

Network File System (NFS) in *Networks and communication management*.

srcmstr Daemon

Purpose

Starts the System Resource Controller.

Syntax

srcmstr /usr/sbin/srcmstr [**-r**] [**-B**]

Description

The **srcmstr** daemon is the System Resource Controller (SRC). The **srcmstr** daemon spawns and controls subsystems, handles short subsystem status requests, passes requests on to a subsystem, and handles error notification.

The **srcmstr** daemon is normally started by using an **inittab** file entry.

Flags

-r Accepts remote requests if the daemon is started with the **-r** flag. If you start **srcmstr** without the **-r** flag, remote requests are ignored.

-B Specifies the **-B** flag that causes the **srcmstr** daemon to run as in previous releases (AIX 4.3.1 and earlier).

Notes:

- The **srcmstr** daemon is typically started from **inittab**. To add the **-r** or **-B** flags, edit **/etc/inittab** and run **init q** or reboot.
- The user must be running as root on the remote system. The local **/etc/hosts.equiv** file or the **/.rhosts** file must be configured to allow remote requests.

Security

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **srcmstr** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Start	Lists in an audit log the name of the subsystems being started.
SRC_Stop	Lists in an audit log the name of the subsystems being stopped.

See Setting Up Auditing in *Security* for more details about how to properly select and group audit events, and how to configure audit event data collection.

Error Recovery

The default **/etc/inittab** specifies the **respawn** flag for the **srcmstr** daemon. If the **srcmstr** daemon terminates abnormally and the **/etc/inittab** specifies the **respawn** flag, the **srcmstr** daemon is restarted. It then determines which SRC subsystems were active during the previous invocation. The daemon re-establishes communication with these subsystems (if it existed previously), and initializes a private kernel extension and the **srcd** daemon to monitor the subsystem processes.

If a subsystem known to the previous invocation of **srcmstr** terminates, the SRC kernel extension notifies the **srcd** daemon. The **srcd** daemon sends a socket message to **srcmstr** and subsystem termination is handled as if the subsystem had been started by the current **srcmstr**. This function can be disabled by

specifying the **-B** flag when the **srcmstr** daemon is started. The SRC kernel extension is in **/usr/lib/drivers/SRC_kex.ext**. The executable for **srcd** is **/usr/sbin/srcd**.

Files

/etc/inittab	Specifies stanzas read by the **init** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Method Object Class.
/etc/hosts.equiv	Specifies that no remote requests will work if the specified host name is not in the **/etc/hosts.equiv** file.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.
/dev/.SRC-unix/SRCD	Specifies the **AF_UNIX** socket file for the **srcd** daemon.
/var/adm/SRC/active_list	Contains a list of active subsystems. **Caution:** The structure of this file is internal to SRC and is subject to change.
/var/adm/SRC/watch_list	Contains a list of subsystem processes active during the previous invocation of the **srcmstr** daemon. **Caution:** The structure of this file is internal to SRC and is subject to change.
/.rhosts	Specifies remote machines and users (root only) that are allowed to request SRC function from this machine.

Related Information

The **auditpr** command, **init** command.

The System resource controller in *Operating system and device management* gives an explanation of subsystems, subservers, and the System Resource Controller.

The Auditing overview in *Security*.

---

startcondresp Command

Purpose

Starts monitoring a condition that has one or more linked responses.

Syntax

To start monitoring a condition:

startcondresp [**–h**] [**–TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

startcondresp {**-U** | **-L**} [**–h**] [**–TV**] *condition*[**:***node_name*] *response*

Description

The **startcondresp** command starts the monitoring of a condition that has a linked response. A link between a condition and a response is called a *condition/response association.* In a cluster environment, the condition and the response must be defined on the same node. After monitoring is started, when the condition occurs, the response is run. If no responses are specified, monitoring is started for all responses

linked to the condition. This causes all of the linked responses to run when the condition occurs. If more than one response is specified, monitoring is started only for those linked responses.

If one or more responses are specified and the responses are not linked with the condition, the **startcondresp** command links the specified responses to the condition, and monitoring is started. Use the **mkcondresp** command to link a response to a condition without starting monitoring.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be started by the **startcondresp** command. If the condition/response association you specify on the **startcondresp** command is locked, it will not be started; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition/response association so it cannot be started, stopped, or removed, reissue this command using its **-L** flag.

Flags

–h Writes the command's usage statement to standard output.

–T Writes the command's trace messages to standard error. For your software service organization's use only.

–V Writes the command's verbose messages to standard output.

–U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

–L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

Parameters

condition Specifies the name of the condition linked to the response. The condition is always specified first.

node_name Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response Specifies the name of one or more responses. Specifying more than one response links the responses to the condition if they are not already linked and starts monitoring for the specified responses.

Security

The user needs write permission for the **IBM.Association** resource class to run **startcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To start monitoring for the condition ″FileSystem space used ″ by using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command:

   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

2. To start monitoring for the condition ″FileSystem space used ″ by using all of its linked responses, run this command:

   ```
   startcondresp "FileSystem space used"
   ```

3. To start monitoring for the condition ″FileSystem space used ″ by using the response ″Broadcast event on-shift″ and ″E-mail root anytime″, whether or not they are linked with the condition, run this command:

   ```
   startcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

These examples apply to management domains:

1. To start monitoring for the condition ″FileSystem space used″ on the management server using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command on the management server:

   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

2. To start monitoring for the condition ″FileSystem space used″ on the managed node **nodeB** using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command on the management server:

   ```
   startcondresp  "FileSystem space used":nodeB "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To start monitoring for the condition ″FileSystem space used″ on **nodeA** in the domain using the response ″Broadcast event on-shift″ (also on **nodeA** in the domain), whether or not the response is linked with the condition, run this command on any node in the domain:

   ```
   startcondresp  "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/startcondresp
Contains the **startcondresp** command

Related Information

Books: *RSCT: Administration Guide* for more information about ERRM operations

Commands: **lscondresp**, **mkcondition**, **mkcondresp**, **mkresponse**, **stopcondresp**

Information Files: **rmccli**

---

start-secldapclntd Command

Purpose

The **start-secldapclntd** script is used to start the **secldapclntd** LDAP client daemon.

Syntax

/usr/sbin/start-secldapclntd [**-C** *CacheSize*] [**-p** *NumOfThread*] [**-t** *CacheTimeOut*] [**-T** *HeartBeatIntv*] [**-o** *ldapTimeOut*]

Description

The **start-secldapclntd** script starts the **secldapclntd** daemon if it is not running. It does not do anything if the **secldapclntd** daemon is already running. The script also cleans the portmapper registration (if there is any) from previous **secldapclntd** daemon process before it starts the **secldapclntd** daemon. This prevents the startup failure of the new daemon process from portmap-per registration failure.

Flags

By default, the **secldapclntd** daemon reads the configuration information specified in the **/etc/security/ldap/ldap.cfg** file at startup. If the following options are given in command line when starting **secldapclntd** process, the options from the command line will overwrite the values in the **/etc/security/ldap/ldap.cfg** file.

-C *CacheSize*
Sets the maximum cache entries used by the **secldapclntd** daemon to CacheSize number of entries. Valid range is 100-10,000 entries for user cache. The default is 1000. The group cache entries will be 10% of the user cache entries.

-o *ldapTimeOut*	Timeout period in seconds for LDAP client requests to the server. This value determines how long the client will wait for a response from the LDAP server. Valid range is 0 - 3600 (1 hour). Default is 60 seconds. Set this value to 0 to disable the timeout and force the client to wait indefinitely.
-p *NumOfThread*	Sets the number of thread used by the **secldapclntd** daemon to **NumOfThread** threads. Valid range is 1-1000. The default is 10.
-t *CacheTimeout*	Sets the cache to expire in CacheTimeout seconds. Valid range is 60- 3600 seconds. The default is 300 seconds.
-T *HeartBeatIntv*	Sets the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.

Examples

1. To start the **secldapclntd** daemon, type:
   ```
   /usr/sbin/start-secldapclntd
   ```
2. To start the **secldapclntd** with using 20 threads and cache timeout value of 600 seconds, type:
   ```
   /usr/sbin/start-secldapclntd -p 20 -t 600
   ```

   It is recommended that you specify these values in the **/etc/security/ldap/ldap.cfg** file, so that these values will be used each time you start the **secldapclntd** process.

Files

/usr/sbin/start-secldapclntd	Used to start the **secldapclntd** LDAP client daemon.

Related Information

The **secldapclntd** daemon

The **mksecldap**, **stop-secldapclntd**, **restart-secldapclntd**, **ls-secldapclntd**, and **flush-secldapclntd** commands.

The **/etc/security/ldap/ldap.cfg** file.

---

stop-secldapclntd Command

Purpose

The **stop-secldapclntd** script is used to terminate the **secldapclntd** LDAP client daemon.

Syntax

/usr/sbin/stop-secldapclntd

Description

The **stop-secldapclntd** script terminates the running **secldapclntd** daemon process. It returns an error if the **secldapclntd** daemon is not running.

Example

To stop the running **secldapclntd** daemon process, type:

```
/usr/sbin/stop-secldapclntd
```

Files

/usr/sbin/stop-secldapclntd Used to terminate the **secldapclntd** LDAP client daemon.

Related Information

The **secldapclntd** daemon

The **mksecldap**, **start-secldapclntd**, **restart-secldapclntd**, **ls-secldapclntd**, and **flush-secldapclntd** commands.

The **/etc/security/ldap/ldap.cfg** file.

---

startrpdomain Command

Purpose

Brings a peer domain that has already been defined online.

Syntax

startrpdomain [**–A** | **–L**] [**–t** *timeout*] [**–Q** *quorum_type* | *quorum_type_name*] [**–m** *fanout*] [**–h**] [**–TV**] *peer_domain*

Description

The **startrpdomain** command brings a defined peer domain online by starting the resources on each node belonging to the peer domain.

The **startrpdomain** command must be run on a node that is defined to the peer domain. The command invites all offline nodes defined to the peer domain to come online in the peer domain every time the command is run for the peer domain. The command can be run more than once in the peer domain. If all the nodes defined in the peer domain are already online, no action is performed.

The **startrpdomain** command determines the peer domain configuration to use to bring the peer domain online by examining the peer domain configuration on the nodes defined to the peer domain. The latest version of the peer domain configuration information that is found is used to bring the peer domain online. By default, the latest version of the peer domain configuration found on at least half of the nodes is used. Specifying the **-A** flag causes the latest version of the peer domain configuration found on all of the nodes defined in the peer domain to be used. Specifying the **-L** flag causes the configuration on the local node to be used.

In determining the latest version of the peer domain configuration information, a configuration timeout defines when to stop checking versions and begin to bring the peer domain online. The default timeout value is 120 seconds. The timeout value can be changed using the **-t** flag. The timeout value should be at least long enough so that the latest version of the peer domain configuration information from at least half of the nodes can be found.

A node can only be online to one peer domain at a time. The **startrpdomain** command cannot be run on a node for a peer domain when another peer domain is already online for that node.

Flags

-A Finds and uses the latest version of the peer domain configuration information from all of the nodes in the peer domain. This flag cannot be specified if the **-L** flag is specified. If neither flag (**-A** or **-L**) is specified, the latest version of the peer domain configuration information from at least half of the nodes in the peer domain is used.

-L Uses the latest version of the peer domain configuration information that is on the local node. This flag cannot be specified if the **-A** flag is specified. If neither flag (**-A** or **-L**) is specified, the latest version of the peer domain configuration information from at least half of the nodes in the peer domain is used.

-t *timeout*
Specifies the timeout value in seconds. This flag limits the amount of time used to find the latest version of the peer domain configuration. When the timeout value is exceeded, the latest version of the peer domain configuration information found thus far is used. The timeout value should be long enough so that the latest version of the peer domain configuration information from at least half of the nodes can be found. The default timeout value is 120 seconds.

-Q *quorum_type* **|** *quorum_type_name*
Enables you to override the startup quorum mode. This can be specified as an integer quorum type or quorum type name. If you do not specify this flag, startup quorum mode will be specified using the **mkrpdomain** command's **-Q** flag (or the default quorum mode for your environment) when you created the peer domain. You can override the quorum startup mode only if the quorum mode has been defined as **normal** or **quick**. The valid values are:

0 | normal
Specifies normal start-up quorum rules. Half of the nodes will be contacted for configuration information.

1 | quick
Specifies quick start-up quorum rules. One node will be contacted for configuration information.

-m *fanout*
Specifies the maximum number of threads to use for this start operation. The **-m** flag overrides the default *fanout* value for the specified peer domain. This value is stored as a persistent attribute in the peer domain's **IBM.PeerNode** class. *fanout* can be an integer from **16** to **2048**.

-h Writes the command's usage statement to standard output.

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

Parameters

peer_domain Specifies the name of a previously-defined peer domain that is to be brought online.

Security

The user of the **startrpdomain** command needs write permission for the **IBM.PeerDomain** resource class on each node that is defined to the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

Restrictions

This command must be run from a node that is defined to the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** is one of the nodes defined to **ApplDomain**.

1. To bring **ApplDomain** online, run this command on **nodeA**:

   ```
   startrpdomain ApplDomain
   ```

2. To bring **ApplDomain** online using all of the nodes in the peer domain to obtain the latest version of the peer domain configuration information, run this command on **nodeA**:

   ```
   startrpdomain -A ApplDomain
   ```

3. To bring **ApplDomain** online using a peer domain configuration timeout value of 240 seconds (to make sure that at least half of the nodes in the peer domain are used), run this command on **nodeA**:

   ```
   startrpdomain -t 240 ApplDomain
   ```

Location

/usr/sbin/rsct/bin/startrpdomain

Contains the **startrpdomain** command

Related Information

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **stoprpdomain**

Information Files: **rmccli**, for general information about RMC-related commands

startrpnode Command

Purpose

Brings one or more nodes online to a peer domain.

Syntax

startrpnode [**–h**] [**–TV**] *node_name1* [*node_name2* ...]

startrpnode –f | **–F** { *file_name* | *"–"* } [**–h**] [**–TV**]

Description

The **startrpnode** command brings one or more offline nodes online to a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The node that is being brought online must have already been defined to be in this peer domain using the **addrpnode** command or the **mkrpdomain** command. The node must not be online to any other peer domain.

Flags

–f | –F { *file_name* **|** *"–"* **}**
: Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

  Use **-f** *"-"* or **-F** *"-"* to specify **STDIN** as the input file.

-h
: Writes the command's usage statement to standard output.

-T
: Writes the command's trace messages to standard error. For your software service organization's use only.

-V
: Writes the command's verbose messages to standard output.

Parameters

node_name1 **[***node_name2* **...]**
: Specifies the peer domain node names of the nodes to be brought online to the peer domain. You can bring one or more nodes online using the **startrpnode** command. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Security

The user of the **startrpnode** command needs write permission for the **IBM.PeerNode** resource class on each node that is to be started in the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0
: The command ran successfully.

1
: An error occurred with RMC.

2
: An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

Restrictions

This command must be run from a node that is online to the peer domain. The node that is to be brought online must be offline to the peer domain, must not be online to any other peer domain, and must be reachable from where the command is run.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f** ″**-**″ or **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In this example, **nodeA** is defined and online to **ApplDomain**, **nodeB** is reachable from **nodeA**, and **nodeB** is not online to **ApplDomain** or any other peer domain. To bring **nodeB** online to **ApplDomain**, run this command from **nodeA**:

```
startrpnode nodeB
```

Location

/usr/sbin/rsct/bin/startrpnode

Contains the **startrpnode** command

Related Information

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpnode**, **preprpnode**, **rmrpnode**, **stoprpnode**

Information Files: **rmccli**, for general information about RMC-related commands

startsrc Command

Purpose

Starts a subsystem, a group of subsystems, or a subserver.

Syntax

To Start a Subsystem

startsrc [**-a** *Argument*] [**-e** *Environment*] [**-h** *Host*] { **-s** *Subsystem* | **-g** *Group*}

To Start a Subserver

startsrc [**-h** *Host*] **-t** *Type* [**-o** *Object*] [**-p** *SubsystemPID*]

Description

The **startsrc** command sends the System Resource Controller (SRC) a request to start a subsystem or a group of subsystems, or to pass on a packet to the subsystem that starts a subserver.

If a start subserver request is passed to the SRC and the subsystem to which the subserver belongs is not currently active, the SRC starts the subsystem and transmits the start subserver request to the subsystem.

Flags

-a *Argument*
Specifies an argument string that is passed to the subsystem when the subsystem is executed. This string is passed from the command line and appended to the command line arguments from the subsystem object class. The *Argument* string specified is a maximum of 1200 characters or the command is unsuccessful. The command argument is passed by the SRC to the subsystem, according to the same rules used by the shell. Quoted strings are passed as a single argument, and blanks outside a quoted string delimit an argument. Single and double quotes can be used.

-e *Environment*
Specifies an environment string that is placed in the subsystem environment when the subsystem is executed. The *Environment* string specified is a maximum of 1200 characters, or the command is unsuccessful. Using the same rules that are used by the shell, the SRC sets up the environment for the subsystem.

Quoted strings are assigned to a single environment variable and blanks outside quoted strings delimit each environment variable to be set. For example: `-e "HOME=/tmp TERM=dumb MESSAGE=\"Multiple word message\""` would set `HOME=/tmp` as the first, `TERM=dumb` as the second, and `MESSAGE="Multiple word message"` as the third environment variable for the subsystem.

-g *Group*
Specifies a group of subsystems to be started. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.

-h *Host*
Specifies the foreign host on which this start action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.

-o *Object*
Specifies that a subserver object is to be passed to the subsystem as a character string. It is the subsystems responsibility to determine the validity of the *Object* string.

-p *SubsystemPID*
Specifies a particular instance of the subsystem to which the start subserver request is to be passed.

-s S*ubsystem*
Specifies a subsystem to be started. The *Subsystem* can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* is not contained in the subsystem object class.

-t *Type*
Specifies that a subserver is to be started. The command is unsuccessful if *Type* is not contained in the subserver object class.

Examples

1. To start a subsystem with arguments and environment variables, enter:

   ```
   startsrc  -s srctest  -a "-D DEBUG"  -e "TERM=dumb HOME=/tmp"
   ```

   This starts the `srctest` subsystem with `"TERM=dumb"`, `"HOME=/tmp"` in its environment and `"-D DEBUG"` as two arguments to the subsystem.
2. To start a subsystem group on a foreign host, enter:

   ```
   startsrc  -g tcpip  -h zork
   ```

   This starts all the subsystems in the subsystem `tcpip` group on the `zork` machine.
3. To start a subserver, enter:

   ```
   startsrc  -t tester
   ```

   This sends a start subserver request to the subsystem that owns the `tester` subsystem.
4. To start a subsystem with command arguments, enter:

   ```
   startsrc  -s srctest  -a "-a 123 -b \"4 5 6\""
   ```

   This places `"-a"` as the first argument, `"123"` as the second, `"-b"` as the third, and`"456"` as the fourth argument to the `srctest` subsystem.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

Related Information

The **stopsrc** command, the **refresh** command.

The System resource controller in *Operating system and device management* gives an explanation of subsystems, subservers, and the System Resource Controller.

---

startup Command

Purpose

Turns on accounting functions at system startup.

Syntax

/usr/sbin/acct/startup

Description

The **startup** command turns on the accounting functions when the system is started, if called by the **/etc/rc** command file. See the **startup** example for the command to add to the **/etc/rc** file.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Examples

To turn on the accounting functions when the system is started, add the following to the **/etc/rc** file:

```
/usr/bin/su - adm -c /usr/sbin/acct/startup
```

The **startup** shell procedure will then record the time and clean up the previous day's records.

Files

/usr/sbin/acct	The path to the accounting commands.

Related Information

The **shutacct** command, **turnacct** command.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in *Operating system and device management*.

Setting up an accounting subsystem in *Operating system and device management* explains the steps you must take to establish an accounting system.

---

startvsd Command

Purpose

startvsd – Makes a virtual shared disk available and activates it.

Syntax

startvsd [–p | **–b**] {**–a** | *vsd_name* ...}

Description

The **startvsd** command makes the specified virtual shared disks available and activates them. It is equivalent to running the **preparevsd** command followed by the **resumevsd** command on the specified virtual shared disk.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Start a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

–p	Specifies that the primary server node defined for the global volume group is to be the active server. See the *RSCT: Managing Shared Disks* for more information.

–b Specifies that the secondary server node defined for the global volume group is to be the active server.

–a Specifies that all virtual shared disks that have been defined are to be started.

Parameters

vsd_name Specifies a virtual shared disk.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To make available and activate the virtual shared disk **vsd1vg1n1**, enter:

```
startvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/startvsd

Related Information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

---

startx Command

Purpose

Initializes an X session.

Syntax

startx [**-d** *Display*:*0*] [**-t** | **-w**] [**-x** Startup | [**-r** *Resources*] [**-m** *Window_Manager*]] [**-wait**]

Description

The **startx** command streamlines the process of starting an X session.

The command does the following:

- Sets the user's **DISPLAY** environment variable to identify the X server to the X clients
- When run from a workstation, starts the X server
- Starts the X clients.

The **startx** command redirects X server and X client error messages to the file specified by the user's **XERRORS** environment variable. This process is useful for debugging and gives the X server a clean startup and shutdown appearance on a workstation.

If a startup script file name is not given at the command line with the **-x** option, then the **startx** command searches for a file specified by the user's **XINITRC** environment variable. If the **XINITRC** environment variable is not set, then the **startx** command searches the user's home directory for a file called **.Xinit**, **.xinit**, **.Xinitrc**, **.xinitrc**, or **.xsession**, respectively, to begin the X client programs.

If a startup file is not found, the **startx** command runs the Window Manager indicated at the command line with the **-m** option, or invokes the window manager **mwm**, **twm**, **awm**, or **uwm** after finding the associated configuration file (**.mwmrc**, **.twmrc** , **.awmrc**, or **.uwmrc**, respectively). If a window manager configuration file is not found in the user's home directory, **startx** initiates an **Xterm** client and the **mwm** window manager.

When a startup file is not found, the **startx** command also instructs the loading of the resources file given at the command line with the **-r** option, or a file from the user's home directory called **.Xdefaults**, **.xdefaults**, **.Xresources**, or **.xresources**, respectively. If an X resources file is not found, then the X session will not be personalized.

If a startup file exists for a workstation and no resources are loaded by the user, then the **xinit** command within the **startx** command attempts to load an **.Xdefaults** file.

The use of a workstation is assumed when the X session is initiated from **/dev/lft***. If this is not the case, then the **-t** or **-w** option must be used.

Flags

-d *Display***:***0*	Specifies the display name of the X server to pass to the X clients during the process for startup.
-m *Window_Manager*	Starts the Window Manager when no startup script is found.
-r *Resources*	Loads the resources file when no startup script is found.
-t	Starts X clients for an X terminal.
-w	Starts the X server and X clients for an X window session on a workstation.
-wait	Prevents the X session from being restarted when the **xdm** command invokes **startx**.
-x Startup	Starts an X window session using the startup script.

Note: You can use one or both of the **-m** and **-r** options, or the **-x** option, but you cannot use the **-x** option with the **-m** and **-r** options. In the startup script, it is the responsibility of the user to start a window manager session, load X resources, and spawn X clients.

Examples

1. To start an X session on a workstation, or an X terminal, enter:

   ```
   startx
   ```

2. To force start an X session on a workstation, enter:
   ```
   startx -w
   ```
3. To start an X session for an X terminal, and log off the user's telnet session, enter:
   ```
   startx; kill -9 $$
   ```
4. To start an X session using the **.xinitrc** script, enter:
   ```
   startx -x .xinitrc
   ```
5. To start an X session using the **mwm** window manager, enter:
   ```
   startx -m mwm
   ```

   However, if a startup script file is found, the **-w** option is ignored.
6. In the startup script, it is the responsibility of the user to start a window manager, load X resources, and spawn X clients. The following is an example of an **.xsession** script.
   ```
   #!/bin/csh
    (mwm &)
    xrdb -load .Xdefaults
    (xclock -g 75x75+0+0 &)
    (xbiff -g 75x75+101-0 &)
    if ("/dev/lft*" == "`tty`") then
       aixterm -g 80x24+0+0 +ut -C -T `hostname`
    else
       aixterm -g 80x24+0+0 +ut -T `hostname`
    endif
   ```

For a workstation, the last line in the startup script should be a foreground **aixterm** command with the **-C** option for console messages.

For an X terminal, the last line in the startup script should be a foreground **aixterm** command without the **-C** option. In addition, because some X terminals do not terminate the **telnet** session upon closing, the user must exit the current telnet session before using hot keys to switch to the X session.

Also, the **startx** command can be used by the **xdm** command in the **/usr/lib/X11/xdm/Xsession** file. This provides the **xdm** command with the features of the **startx** command.

Files

The following file names have been historically used for the startup of an X session.

$HOME/.xerrors	Where **startx** is to redirect error messages. By default, **startx** redirects errors to the **.xerrors** file in user's home directory.
$HOME/.Xinit,** **$HOME/.xinit, **$HOME/.Xinitrc**, **$HOME/.xinitrc**,	
$HOME/.xsession	Used as a Startup file containing shell commands to start a window manager, load X resources, and spawn X clients.
$HOME/.Xdefaults,** **$HOME/.xresources	Used as an X resources file loaded to set user preferences for X clients.
$HOME/.mwmrc	An **mwm** configuration file.
$HOME/.twmrc	A **twm** configuration file.
$HOME/.awmrc	An **awm** configuration file.
$HOME/.uwmrc	A **uwm** configuration file.
/dev/lft*	The terminal, or tty, interface of a workstation's initial **login** shell.

Related Information

The **mwm** command, **xinit** command, **xdm** command, **aixterm** command, **telnet, tn, or tn3270** command, **X** command, and **xrdb** command.

---

statd Daemon

Purpose

Provides crash and recovery functions for the locking services on NFS.

Syntax

/usr/sbin/rpc.statd [**-t** *threads*]

Description

The **statd** daemon interacts with the **lockd** daemon to provide crash and recovery functions for the locking services on Network File System (NFS). The **statd** daemon should always be started before the **lockd** daemon.

The **statd** daemon is started and stopped by the following SRC commands:

```
startsrc -s rpc.statd
stopsrc -s rpc.statd
```

The status monitor maintains information on the location of connections as well as the status in the **/var/statmon/sm** directory, the **/var/statmon/sm.bak** directory, and the **/var/statmon/state** file. When restarted, the **statd** daemon queries these files and tries to reestablish the connection it had prior to termination. To restart the **statd** daemon, and subsequently the **lockd** daemon, without prior knowledge of existing locks or status, delete these files before restarting the **statd** daemon.

Flags

-t *threads*	Specifies the maximum number of rpc.statd threads allowed. The Default value is 50.

Related Information

The **lockd** daemon.

List of NFS commands.

Network File System (NFS) in *Operating system and device management*.

---

statvsd Command

Purpose

statvsd – Displays virtual shared disk device driver statistics of a node.

Syntax

statvsd

Description

The **statvsd** command displays virtual shared disk statistics of a node. For example, on a busy server an increasing number of ″requests queued waiting for a buddy buffer″ is normal and does not necessarily

imply a problem. Of more value is the ″average buddy buffer wait_queue size″ which is the number of requests queued for a buddy buffer when the **statvsd** command was issued. See the ″Examples″ section for the meaning of output lines.

Flags

None.

Parameters

None.

Security

You must be in the AIX **bin** group to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

The following examples display virtual shared disk device driver statistics.

1. The header line indicates the version and release of the code. For example:

   ```
   VSD driver (vsdd):  IP/SMP Version:4 Release:1
   ```

2. The level of virtual shared disk parallelism defaults to 9 and is the buf_cnt parameter on the uphysio call that the device driver makes in the kernel. For example:

   ```
   9 vsd parallelism
   ```

3. The maximum IP message size in bytes. For example:

   ```
   61440 vsd max IP message size
   ```

4. The number of requests that had to wait for a request block. For example:

   ```
   61440 vsd max IP message size
   ```

5. The number of requests that had to wait for a pbuf (a buffer used for the actual physical I/O request submitted to the disk). For example:

   ```
   0 requests queued waiting for a pbuf
   ```

6. The number of requests that had to wait for a buddy buffer. A buffer that is used on a server to temporarily store date for I/O operations originating at a client node. For example:

   ```
   2689 requests queued waiting for a buddy buffer
   ```

7. The number of requests queued for a buddy buffer when the **statvsd** command was issued. For example:

   ```
   0 average buddy buffer wait_queue size
   ```

8. The number of requests that a server has rejected, typically because of an out-of-range sequence number or an internal problem. For example:

```
4 rejected requests
```

9. The number of responses that a client has rejected. Typically because a response arrived after a retry was already sent to the server. For example:

   ```
   0 rejected responses
   ```

10. The number of requests that were placed on the rework queue. For example:

    ```
    0 requests rework
    ```

11. The number of read requests that were not on a 64 byte boundary. For example:

    ```
    0 64 byte unaligned reads
    ```

12. The number of requests that got a DMA shortage. This condition would require the I/O operation to be executed in nonzero copy mode. For example:

    ```
    0 DMA space shortage
    ```

13. The number of requests that have timed out. The current timeout period is approximately 15 minutes. For example:

    ```
    0 timeouts
    ```

14. There are a fixed number of retries. The retries counters display the number of requests that have been retried for that particular ″retry bucket.″ Numbers appearing further to the right represent requests that have required more retries. When a request exhausts its number of retries, it gets recorded as a timeout. For example:

    ```
    retries:  0 0 0 0 0 0 0 0 0

              0 total retries
    ```

15. Sequence numbers are internally used by the device driver. These numbers are managed by the device driver and the Recoverable virtual shared disk subsystem. For example:

    ```
    Non-zero Sequence Numbers

    node#     expected     outgoing     outcase?     Incarnation:0
     11        125092         0            |

     11 Nodes Up with zero sequence numbers: 1 3 5 7 9 11 12 13 14 15 16
    ```

Location

/opt/rsct/vsd/bin/statvsd

Related Information

Commands: **ctlvsd**, **vsdnode**

Refer to *RSCT: Managing Shared Disks* for information on tuning virtual shared disk performance.

---

stopcondresp Command

Purpose

Stops the monitoring of a condition that has one or more linked responses.

Syntax

To stop monitoring a condition:

stopcondresp [**–q**] [**–h**] [**–TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

stopcondresp {**-U** | **-L**} [**–h**] [**–TV**] *condition*[**:***node_name*] *response*

Description

The **stopcondresp** command stops the monitoring of a condition that has one or more linked responses. If no response is specified, all of the linked responses for the condition are stopped. If one or more responses is specified, only those responses that are linked to the condition are stopped. When the condition occurs, the response is not run. If no responses are active for a condition, the condition is no longer monitored.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be stopped by the **stopcondresp** command. If the condition/response link you specify on the **stopcondresp** command is locked, it will not be stopped; instead an error will be generated informing you that the condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. A condition/response association is typically locked because it is essential for system software to work properly, so you should exercise caution before unlocking it.

Flags

–q Does not return an error when either *condition* or *response* does not exist or when the *condition* linked with *response* is not being monitored.

–h Writes the command's usage statement to standard output.

–T Writes the command's trace messages to standard error. For your software service organization's use only.

–V Writes the command's verbose messages to standard output.

–U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

–L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

Parameters

condition Specifies the name of the condition linked to the response. The condition is always specified first.

node_name Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response Specifies the names of one or more responses. Monitoring is stopped for the specified responses. (If a specified response is not linked to the condition, it is ignored.)

Security

The user needs write permission for the **IBM.Association** resource class to run **stopcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To stop monitoring for the condition ″FileSystem space used ″ which has the response ″Broadcast event on-shift″ linked with it, run this command:

   ```
   stopcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To stop monitoring for the condition ″FileSystem space used ″ using all of its linked responses, run this command:

   ```
   stopcondresp  "FileSystem space used"
   ```

This example applies to management domains:

1. To stop monitoring for the condition ″FileSystem space used ″ on the managed node **nodeB** which has the response ″Broadcast event on-shift″ linked with it, run this command on the management server:

   ```
   stopcondresp "FileSystem space used:nodeB" "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To stop monitoring for the condition ″FileSystem space used ″ on the node **nodeA** which has the response ″Broadcast event on-shift″ linked with it, run this command on any node in the domain:

   ```
   stopcondresp "FileSystem space used:nodeA" "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/stopcondresp
Contains the **stopcondresp** command

Related Information

Books: *RSCT: Administration Guide*, for more information about ERRM operations

Commands: **lscondresp**, **mkcondition**, **mkcondresp**, **mkresponse**, **startcondresp**

Information Files: **rmccli**

---

stoprpdomain Command

Purpose

Takes an online peer domain offline.

Syntax

stoprpdomain [**–f**] [**–h]** [**–TV**] *peer_domain*

Description

The **stoprpdomain** command takes all of the nodes that are currently online in the peer domain offline. The peer domain definition is not removed from the nodes.

The command must be run on a node that is online in the peer domain. If the command is run on a node that is offline to the peer domain, no action is performed.

The **-f** flag must be used to override a subsystems rejection of the request to take the peer domain offline. A subsystem may reject the request if a peer domain resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystems that the peer domain must be brought offline regardless of the resource state.

Flags

-f Forces the subsystems to accept the stop request when it otherwise would not.

-h Writes the command's usage statement to standard output.

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

Parameters

peer_domain Specifies the name of the online peer domain that is to be brought offline.

Security

The user of the **stoprpdomain** command needs write permission for the **IBM.PeerDomain** resource class on each node that is defined to the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f** ″**-**″ or **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** is one of the nodes defined and is online to **ApplDomain**.

1. To take **ApplDomain** offline, run this command on **nodeA**:

   ```
   stoprpdomain ApplDomain
   ```

2. To take **ApplDomain** offline while making sure the stop request will not be rejected by any subsystem, run this command on **nodeA**:

   ```
   stoprpdomain -f ApplDomain
   ```

Location

/usr/sbin/rsct/bin/stoprpdomain
Contains the **stoprpdomain** command

Related Information

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **startrpdomain**

Information Files: **rmccli**, for general information about RMC-related commands

---

stoprpnode Command

Purpose

Takes one or more nodes offline from a peer domain.

Syntax

stoprpnode [**–f**] [**–h**] [**–TV**] *node_name1* [*node_name2*...]

stoprpnode –F { *file_name* | *"–"* } [**–f**] [**–h**] [**–TV**]

Description

The **stoprpnode** command takes an online node offline from a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The **-f** flag must be used to override a subsystem's rejection of the request to take a node offline. A subsystem may reject the request if a node resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystems that the node must be brought offline regardless of the resource state.

If this command is used to take more than one node offline by specifying more than one *node_name* parameter, and the node that this command is running on is in the list, it will be brought offline last.

Flags

-f Forces the subsystems to accept the stop request when it otherwise would not.

–F { *file_name* **|** *"–"* **}**
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

Use **-F** *"-"* to specify **STDIN** as the input file.

-h Writes the command's usage statement to standard output.

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

Parameters

node_name1 **[***node_name2***...]** Specifies the peer domain node names of the nodes that are to be

brought offline from the peer domain. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Security

The user of the **stoprpnode** command needs write permission for the **IBM.PeerNode** resource class on each node that is to be stopped in the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

Restrictions

This command must be run on a node that is online to the peer domain. The node to be brought offline must be reachable from the node on which the command is run.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** and **nodeB** are online to **AppIDomain**.

1. To take **nodeB** offline, run this command on **nodeA**:

```
stoprpnode nodeB
```

2. To take **nodeB** offline and force the offline request, run this command on **nodeA**:
   ```
   stoprpnode -f nodeB
   ```

Location

/usr/sbin/rsct/bin/stoprpnode
Contains the **stoprpnode** command

Related Information

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpnode**, **preprpnode**, **rmrpnode**, **startrpnode**

Information Files: **rmccli**, for general information about RMC-related commands

---

stopsrc Command

Purpose

Stops a subsystem, a group of subsystems, or a subserver.

Syntax

To Stop a Subsystem

stopsrc [**-h** *Host*] [**-f** | **-c**] { **-a** | **-g** *Group* | **-p** *SubsystemPID* | **-s** *Subsystem* }

To Stop a Subserver

stopsrc [**-h** *Host*] [**-f**] **-t** *Type* [**-p** *SubsystemPID*] [**-P** *SubserverPID* | **-o** *Object*]

Description

The **stopsrc** command sends a request to the System Resource Controller (SRC) to stop a subsystem, a group of subsystems, or all subsystems. The **stopsrc** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem for a stop subserver request.

In the absence of the **-f** (stop force) flag, a normal stop action is assumed. A normal stop requests that a subsystem or subserver complete all current processing, release resources when all application activity has been completed, and then end. No new requests for work should be accepted by the subsystem.

A forced stop requests that a subsystem or subserver end quickly, releasing all resources, but not wait for application activity to complete.

A cancel action stops the subsystem after the subsystem's resources are released and after a grace period. This grace period is specified in the subsystem object class. The cancel stop is used only for subsystem stops and is always sent to the subsystem as the **SIGTERM** signal. The subsystem should catch this signal, perform subsystem clean up operations, and end. If the subsystem does not end within the wait time period, specified in the subsystem object class, the subsystem is sent a **SIGKILL** signal to ensure that the subsystem stops.

If the subsystem uses sockets or message queues for communication, a packet is constructed and sent to the subsystem. If the subsystem uses signals for communication, the subsystem is sent the appropriate signal from the subsystem object class.

Flags

-a	Specifies that all subsystems are to be stopped.
-c	Specifies that the stop request is a canceled stop request. For a cancel stop request, a **SIGTERM** signal is sent to the subsystem. After the wait time contained in the subsystem object class has passed, if the subsystem has not yet ended, the subsystem is sent a **SIGKILL** signal.
-f	Specifies a forced stop request.
-g *Group*	Specifies that a group of subservers is to be stopped. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign *Host* machine on which this stop action is requested. The local user must be running as ″root″. The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-o *Object*	Specifies that a subserver *Object* value is to be passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to stop, or a particular instance of the subsystem to which the stop subserver request is to be passed.
-P *SubserverPID*	Specifies that a subserver PID is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to be stopped. The *Subsystem* parameter can be the actual subsystem name or the synonym name for the subsystem. The **stopsrc** command stops all currently active instances of the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies that a subserver is to be stopped. The **stopsrc** command is unsuccessful if the *Type* specified is not contained in the subserver object class.

Examples

1. To stop force a subsystem on a foreign host, enter:

   ```
   stopsrc  -h zork  -s srctest  -f
   ```

   This forces a stop on all the instances of the `srctest` subsystem on the `zork` machine.
2. To stop cancel a subsystem group, enter:

   ```
   stopsrc  -g tcpip  -c
   ```

   This activates a stop cancel on all the subsystems in the `tcpip` group.
3. To stop a subserver, enter:

   ```
   stopsrc  -t tester  -p 1234
   ```

   This stops the `tester` subserver that belongs to the `srctest` subsystem with a subsystem PID of `1234`.
4. To stop all subsystems, enter:

   ```
   stopsrc  -a
   ```

   This stops all the active subsystems on the local machine.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.

/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

Related Information

The **startsrc** command, the **refresh** command.

The System resource controller in *Operating system and device management* gives an explanation of subsystems, subservers, and the System Resource Controller.

---

stopvsd Command

Purpose

stopvsd – Makes a virtual shared disk unavailable.

Syntax

stopvsd {**–a** | *vsd_name* ...}

Description

The **stopvsd** command brings the specified virtual shared disks from the suspended state to the stopped state. This makes the virtual shared disks unavailable. All applications that have outstanding requests for the virtual shared disk see these requests terminate with error. Read and write requests return errors with **errno** set to **ENODEV**. If the virtual shared disk is in the stopped state, this command leaves it in the stopped state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Stop a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

–a	Specifies that all virtual shared disks in the suspended state are to be stopped.

Parameters

vsd_name	Specifies a virtual shared disk. If the virtual shared disk is not is the suspended state, you get an error message.

Security

You must have root authority to run this command.

Exit Status

0	Indicates the successful completion of the command.
nonzero	Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the suspended state to the stopped state, enter:

```
stopvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/stopvsd

Related Information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startvsd**, **suspendvsd**, **ucfgvsd**

---

stpinet Method

Purpose

Disables the inet instance.

Syntax

stpinet [**-l** *"Interface ..."*] [**-t** *Time*]

Description

If **stpinet** is started with a list of network interfaces specified with the **-l** option, then this method only stops those IFs. Otherwise, **stpinet** informs users of the impending demise of TCP/IP, using the **wall** command, and invokes the **ifconfig** command to mark each configured IF as **down**. If no network interfaces are specified, the status flag of the inet instance is set to DEFINED.

Flags

-l *"Interface ..."*	Specifies the name of the interface to be disabled.
-t *Time*	Specifies the time in minutes until the inet instance is stopped.

Examples

The following example disables the inet instance `tr0` five minutes from the time the method is executed:

```
stpinet -l "tr0" -t 5
```

Related Information

The **ifconfig** command, **rmdev** command, **wall** command.

The **odm_run_method** subroutine.

Writing a Device Method in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

TCP/IP network interfaces in *Networks and communication management*.

---

strace Command

Purpose

Prints STREAMS trace messages.

Syntax

strace [*mid sid level*] ...

Description

The **strace** command without parameters writes all STREAMS event trace messages from all drivers and modules to its standard output. These messages are obtained from the STREAMS log driver. If parameters are provided, they must be in triplets. Each triplet indicates that tracing messages are to be received from the given module or driver, subID (usually indicating minor device), and priority level equal to or less than the given level. The `all` token may be used for any member to indicate no restriction for that attribute.

Parameters

mid	Specifies a STREAMS module ID number.
sid	Specifies a subID number.
level	Specifies a tracing priority level.

Output Format

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 system was started
`<level>`	Tracing priority level
`<flags>`	Has one of the following values: **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>`	SubID number of source

<text> Formatted text of the trace message

On multiprocessor systems, <text> is composed of two parts:

- the number of the processor where the owner of the message has sent it,
- the formatted text itself.

Once initiated, the **strace** command continues to execute until terminated by the user.

> **Note:** Due to performance considerations, only one **strace** command 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 have a greater impact on overall STREAMS performance. Running the **strace** command has 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.

Examples

1. To output all trace messages from the module or driver whose module ID is 41, enter:

   ```
   strace 41 all all
   ```

2. To output those trace messages from driver or 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.

Related Information

List of Streams Commands, STREAMS Overview, Understanding the log Device Driver in *AIX 5L Version 5.3 Communications Programming Concepts*.

---

strchg Command

Purpose

Changes stream configuration.

Syntax

To push modules onto a stream:

strchg -h *Module1* [**,** *Module2* ...]

To pop modules off a stream:

strchg -p [**-a** | **-u** *Module*]

To push and pop modules to conform to the configuration file:

strchg -f *File*

Description

The **strchg** command is used to alter the configuration of the stream associated with the user's standard input. The **strchg** command pushes modules on the stream, pops modules off of the stream, or both. Only

the root user or owner of a STREAMS device can alter the configuration of that stream. If another user attempts to alter the configuration, the **strchg** command will not succeed.

Note: If modules are pushed in the wrong order, the stream might not function as expected.

Flags

-a Pops all modules above the topmost driver off of a stream. The **-p** flag must be used in front of the **-a** flag.

-f *File* Pushes and pops the necessary modules to conform the stream to the configuration given in the specified file.

The **-h**, **-p**, and **-f** flags are mutually exclusive.

-h *Module1* Pushes modules onto a stream. The modules are listed on the command line in the order they are to be pushed.

-p Pops a module off of a stream. Used alone, the **-p** flag pops the topmost module from the stream.

-u *Module* Pops all modules above the specified module off of a stream. The **-p** flag must be used in front of the **-u** flag.

The **-a** and **-u** flags are mutually exclusive.

Parameters

Module1 Specifies the module to be pushed onto a stream. (Used by the **-h** flag.)

Module Specifies the topmost module to remain on a stream. All modules above this module are popped off of the stream. (Used by the **-u** flag.)

File Contains a list of modules representing the desired configuration of the stream. Each module name must appear on a separate line, where the first name represents the topmost module and the last name represents the module that is closest to the driver.

Return Values

On successful completion, the **strchg** command returns a value of 0. Otherwise, it returns a nonzero value and prints an error message indicating usage error, a bad module name, too many modules to push, failure of an **ioctl** operation on the stream, or failure to open the file specified by the *File* parameter.

Examples

1. To push the `ldterm` module on the stream, enter:

   ```
   strchg -h ldterm
   ```

2. To pop the topmost module from the stream associated with the `/dev/term/24` device, enter:

   ```
   strchg -p < /dev/term/24
   ```

   The user must be the owner of this device or the root user.

3. If the `fileconf` file contains the following:

   ```
   compat
   ldterm
   ptem
   ```

   the following command configures the stream so that the `ptem` module is pushed over the driver, followed by the `ldterm` module, and the `compat` module is pushed closest to the stream head.

   ```
   strchg -f fileconf
   ```

Related Information

The **strconf** command.

List of Streams Commands, STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

The **streamio** operations in *AIX 5L Version 5.3 Technical Reference: Communications Volume 2*.

---

strclean Command

Purpose

Cleans up the STREAMS error logger.

Syntax

strclean [**-d**] [**-a** *Age*]

Description

The **strclean** command is used to clean up the STREAMS error-logger directory on a regular basis: for example, by using the **cron** daemon. By default, all files with names matching **error.*** in the **/var/adm/streams** directory that have not been modified in the last three days are removed.

Note: The **strclean** command is typically run using the **cron** deamon on a daily or weekly basis.

Flags

-a *Age*	Specifies the maximum age, in days, for a log file.
-d	Specifies a directory other than the default directory.

Examples

The following example has the same result as running the **strclean** command with no parameters.

```
strclean -d /var/adm/streams -a 3
```

Files

/var/adm/streams/error.*	Contains the STREAMS error log.

Related Information

The **cron** daemon, and **strerr** daemon.

List of Streams Commands, STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

---

strconf Command

Purpose

Queries stream configuration.

Syntax

strconf [**-t** | **-m** *module*]

Description

The **strconf** command is used to query the configuration of a stream. When used without any flags, it prints a list of all the modules in the stream as well as the topmost driver. The list is printed with one name per line, where the first name printed is the topmost module on the stream and the last item printed is the name of the driver.

Note: The **strconf** command only reads from standard input.

Flags

-m *Module* Determines if the specified module is present on the stream. If the module is present, the **strconf** command prints the message `yes` and returns a value of 0. If it is not present, the **strconf** command prints the message `no` and returns a nonzero value.

The **-t** and **-m** flags are mutually exclusive.

-t Prints only the topmost module of the stream (if one exists).

Parameter

Module Specifies the module for which to look.

Examples

1. For a stream that has only the `ldterm` module pushed above the `ports` driver, the **strconf** command (with no flags) would produce the following output:

   ```
   ldterm
   ports
   ```

2. Entering the following command asks if the `ldterm` module is on the stream:

   ```
   strconf -m ldterm
   ```

   The command produces the following output while returning an exit status of 0:

   ```
   yes
   ```

Related Information

The **strchg** command.

The **streamio** operations in *AIX 5L Version 5.3 Technical Reference: Communications Volume 1*.

List of Streams Commands, STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

---

strerr Daemon

Purpose

Receives error log messages from the STREAMS log driver.

Syntax

strerr

Description

The **strerr** daemon receives error log messages from the STREAMS log driver and appends them to a log file. The error log files produced reside in the directory **/var/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> <sud> <text>
```

These fields are defined as follows:

<seq>	Error sequence number
<time>	Time of message in *hh:mm:ss*
<ticks>	Time of message in machine ticks since boot priority level
<flags>	Has one of the following values:
	T The message was also sent to a tracing process
	F Indicates a fatal error
	N Send mail to the person who administers your system
<mid>	Module ID number of source
<sid>	Sub-ID number of source
<text>	Formatted text of the error message

On multiprocessor systems, <text> is composed of two parts:

- the number of the processor where the owner of the message has sent it,
- the formatted text itself.

Messages that appear in the error log are intended to report exceptional conditions that require the attention of the person who administers your system. Those messages indicating the total failure of a STREAMS driver or module should have the **F** flag set. Those messages requiring the immediate attention of the administrator should have the **N** flag set, which causes the error logger to send the message to that person by way of the **mail** command. The priority level usually has no meaning in the error log, but does have meaning if the message is also sent to a tracer process.

Once initiated, the **strerr** daemon continues to execute until terminated by the user. Usually, the **strerr** daemon is executed asynchronously.

> **Note:** Only one **strerr** daemon 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 causes a degradation in STREAMS performance. If a large number of messages are generated 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.

Files

/var/adm/streams/error.*mm-dd*	Error log file.

Related Information

List of Streams Commands, STREAMS Overview**,** Understanding the log Device Driver in *AIX 5L Version 5.3 Communications Programming Concepts*.

strinfo Command

Purpose

Displays administrative information about STREAMS activity.

Syntax

strinfo -m | -q

Description

The **strinfo** command displays information for debugging purposes about STREAMS, drivers and modules, or stream heads and the STREAMS run queue.

Flags

-m Displays information on drivers and modules present in STREAMS.

-q Displays informations on active stream heads, and on the run queue which holds the STREAMS module and driver service procedures.

Examples

1. To display information about STREAMS drivers and modules in use, enter:

   ```
   strinfo  -m
   ```

   This produces a listing similar to the following:

   ```
   Device: 'sad', dcookie 0xf, flags:0x4, str 0x19a69e8
   Device: 'slog', dcookie 0x10, flags:0x4, str 0x19a6c18
   Device: 'rs', dcookie 0x11, flags:0x2, str 0x19bcb00
   Module: 'bufcall', flags:0x1, str 0x19a5c00
   Module: 'ldterm', flags:0x0, str 0x19cc858
   ```

   In this example `dcookie` indicates the major number, `flags` indicates the flags configuration, and `str` is the STREAMS table address.

2. To display information about active stream heads and the STREAMS run queue, enter:

   ```
   strinfo  -q
   ```

   This produces a listing similar to the following:

   ```
   Active Stream Heads
   sth      sth_dev  sth_rq   sth_wq   sth_flag rq->q_first
   05a7ee00 00110001 05ad7000 05ad7074 00000818 00000000

   STREAMS Service Queue
   Queue 0x5ad7000 Flags 0x10
   ```

File

/usr/sbin/strinfo	Contains the **strinfo** command.

Related Information

List of Streams Commands in *AIX 5L Version 5.3 Communications Programming Concepts*.

STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

strings Command

Purpose

Finds the printable strings in an object or binary file.

Syntax

strings [**-a**] [**-**] [**-o**] [**-t** *Format*] [**-n** *Number*] [**-***Number*] [*File* ...]

Description

The **strings** command looks for printable strings in an object or binary file. A string is any sequence of 4 or more printable characters that end with a new-line or a null character. The **strings** command is useful for identifying random object files.

Flags

-a or **-** Searches the entire file, not just the data section, for printable strings. If this flag is omitted, the **strings** command only looks in the initialized data space of object files.

-n *Number* Specifies a minimum string length other than the default of 4 characters. The maximum value of a string length is 4096. This flag is identical to the **-***Number* flag.

-o Lists each string preceded by its octal offset in the file. This flag is identical to the **-t o** flag.

-t *Format* Lists each string preceded by its offset from the start of the file. The format is dependent on the character used as the *Format* variable.

- **d** Writes the offset in decimal.
- **o** Writes the offset in octal.
- **x** Writes the offset in hexadecimal.

Note: When the **-o** and the **-t** *Format* flags are defined more than once on a command line, the last flag specified controls the behavior of the **strings** command.

-*Number* Specifies a minimum string length other than the default of 4 characters. The maximum value of a string length is 4096. This flag is identical to the **-n** *Number* flag.

File Binary or object file to be searched.

Exit Status

This command returns the following exit values:

0 Specifies that the command ran successfully.

>0 Specifies that an error occurred.

Examples

1. To search a file, enter:

```
strings strings
```

The **string** command displays:

```
@(#)56
1.17 com/cmd/scan/strings.c, cdmscan, bos320 5/7/92 10:21:20
Standard input
strings.cat
/usr/mbin/strings
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
```

```
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
%7o
%7d
%7x
%7o
%7d
```

2. To search for strings at least 12 characters long, enter:

```
strings -12 strings
```

   The **string** command displays:

```
1.17 com/cmd/scan/strings.c, cdmscan, bos320 5/7/92 10:21:20
Standard input
/usr/mbin/strings
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
```

3. To search for strings at least 20 characters long and show the offset in hexadecimal, enter:

```
strings -t x -n 20 strings
```

   The **string** command displays:

```
1017 1.17 com/cmd/scan/strings.c, cmdscan, bos320 5/7/92 10:21:20
108c Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
10d8 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
1124 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
1170 Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
11bc Usage: strings [-a | -] [-o] [-t format] [-n | -#] [file...]
```

Related Information

The **od** command.

---

strip Command

Purpose

Reduces the size of an Extended Common Object File Format (XCOFF) object file by removing information used by the binder and symbolic debug program.

Syntax

strip [**-V**] [**-r** [**-I**] | **-x** [**-I**] | **-t** | **-H** | **-e** | **-E**] [**-X** {**32**|**64**|**32_64**}] [**—**] *File* ...

Description

The **strip** command reduces the size of XCOFF object files. The **strip** command optionally removes the line number information, relocation information, the debug section, the typchk section, the comment section, file headers, and all or part of the symbol table from the XCOFF object files. Once you use this command, symbolic debugging of the file is difficult; therefore, you should normally use the **strip** command only on production modules that you have debugged and tested. Using the **strip** command reduces the storage overhead required by an object file.

For each object module, the **strip** command removes information as specified by the supplied options. For each archive file, the **strip** command removes the global symbol table from the archive.

You can restore a stripped symbol table to an archive or library file by using the **ar -s** command.

The **strip** command with no options removes the line number information, relocation information, symbol table, the debug section, and the typchk section, and the comment section.

Flags

-e Sets the **F_LOADONLY** flag in the optional header of the object file. If the object file is placed in an archive, this flag indicates to the binder (**ld** command) that symbols in the object file should be ignored when linking with the archive.

-E Resets (turns off) the **F_LOADONLY** bit in the optional header of the object file. (See **-e** flag).

-H Removes the object file header, any optional header, and all section headers.

Note: Symbol Table information is not removed.

-l (Lowercase L) Strips the line number information from the object file.

-r Removes all symbol table information except those entries for external and static symbols. Does not remove the relocation information. Also removes the debug and typchk sections. This option produces an object file that can still be used as input to the linkage editor (**ld** command).

-t Removes most symbol table information but does not remove function symbols or line number information.

-V Prints the version number of the **strip** command.

-x Removes the symbol table information but does not remove static or external symbol information. The **-x** flag also removes relocation information, therefore linking to the file would not be possible.

-X *mode* Specifies the type of object file **strip** should examine. The *mode* must be one of the following:

32 Processes only 32-bit object files

64 Processes only 64-bit object files

32_64 Processes both 32-bit and 64-bit object files

The default is to process 32-bit object files (ignore 64-bit objects). The *mode* can also be set with the **OBJECT_MODE** environment variable. For example, **OBJECT_MODE=64** causes **strip** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

— (Double hyphen) Interprets all arguments following this flag as file names. This allows you to strip files whose names start with a hyphen.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To remove the symbol table and line number information from the **a.out** file, enter:
   ```
   strip a.out
   ```
2. To remove the object file header of the **a.out** file, enter:
   ```
   strip -H a.out
   ```
3. To remove both the 32-bit and 64-bit symbol tables from **lib.a**, enter:
   ```
   strip -X 32_64 lib.a
   ```

Files

/usr/ccs/bin/strip Contains the **strip** command.

Related Information

The **ar** command, **as** command, **dump** command, **ld** command, **size** command.

The **ar** file, **a.out** file.

---

stripnm Command

Purpose

Displays the symbol information of a specified object file.

Syntax

stripnm [**-x** | **-d**] [**-s**] [**-z**] *File*

Description

The **stripnm** command (when run without the **-s** flag) prints the symbol table of a specified object file to standard output. The file specified by the *File* parameter can be a single object file or an archive library of object files. If the file specified by the *File* parameter is an archive, a listing for each object file in the archive is produced. If the symbol table has been stripped from the object file, the **stripnm** command extracts symbol names from the traceback tables (even if the **-s** flag is not specified) and the loader section of the object file(s). If the traceback tables do not exist, an error message is displayed.

Each symbol name is preceeded by its address and one character representing the symbol type (similar to **nm** output). When used with **-z**, the output format is the same as it was before AIX 5.2, that is each symbol name is followed by its address (a series of blanks if the address is undefined) and the type of class and section type. The address field can be displayed as a decimal (the default value with **-z**, or when **-d** is used) or hexadecimal (the default value without **-z**, or if the **-x** flag is used).

Source file names are also collected and reported by the **stripnm** command. All the symbols following a source file name line belongs to the same source file, until the next source file name line is encountered. For stripped files, the source file name is reported as being the object file name.

When run using the **-s** flag, the **stripnm** command ignores the symbol table if present and always extracts routine names from the traceback tables and the loader section of the object file(s).

When no symbol table is present or the **-s** flag is used, the **stripnm** command also searches for glue code and pointer glue information. Both are sequences of instructions found in the text section of the object file.

The glue code for 32 bit applications is composed of the following sequences of instructions:

```
8182xxxx #   lwz r12,xxxx(r12) (xxxx is the TOC entry index)
90410014 #   stw r2,14(r1)
800c0000 #   lwz r0,0(r12)
804c0004 #   lwz r2,4(r12)
7c0903a6 # mtctr r0
4e800420 # bctr
```

The loader section entry whose address matches the TOC entry pointed to by xxxx gives the function name for this sequence of glue code.

For 64 bit executables, the glue code sequences are as follows:

```
982xxxx #    ld r12,xxxx(r2) (xxxx is the TOC entry index)
8410028 #   std r2,28(r1)
80c0000 #    ld r0,0(r12)
```

```
84c0008 #    ld r2,8(r12
c0903a6 # mtctr r0
e800420 #  bctr
```

The pointer glue code for 32 bit applications is composed of the following sequence:

```
800b0000 #   lwz r0,0(r11)
90410014 #   stw r2,20(r1)
7c0903a6 # mtctr r0
804b0004 #   lwz r2,4(r11)
816b0008 #   lwz r11,8(r11)
4e80xx20 #  bctr
```

For 64bit executables, the pointer glue code sequence is as follows:

```
e80b0000 #    ld r0,0(r11)
f8410028 #   std r2,20(r1)
7c0903a6 # mtctr r0
e84b0008 #    ld r2,8(r11)
e96b0010 #    ld r11,16(r11)
4e80xx20 #  bctr
```

Pointer glue exists only in one copy and is always reported as symbol `._prtgl`.

The stripnm command searches the Text section from beginning to end for these sequences. If the command finds a sequence of instructions that matches, it is reported as glue code or pointer glue.

Source file symbols are generated artificially by **stripnm** for both glue code and pointer glue. For 32 bit executables, the source file is `glink.s` for all glue code entries, and `ptrgl.s`, for the pointer glue. For 64 bit executables, the source files are repectively `glink64.s` and `ptrgl_64.s`.

The **stripnm** command can also be used to search for symbol information in the **/unix** file. If the **/unix** file does not correspond to the currently running kernel, a warning message displays.

Flags

-d	Prints symbol address values in decimal format. This is the default with **-z**.
-s	Forces to ignore symbol table.
-x	Prints symbol address values in hexadecimal format. This is the default without **-z**.
-z	Uses the old format.

Examples

1. To list the symbols of the **a.out** object file, type:

   ```
   stripnm a.out
   ```

2. To list the symbols address values, in decimal, from the **a.out** object file, type:

   ```
   stripnm -d a.out
   ```

3. To list symbols from the object file from libc.a in the old format, but using hexadecimal addresses, type:

   ```
   stripnm -xz libc.a
   ```

Related Information

The **strip** command.

strload Command

Purpose

Loads and configures Portable Streams Environment (PSE).

Syntax

strload [**-u** | **-q**] [**-f** *File*] [**-d** *List*] [**-m** *List*]

Description

The **strload** command enables the system administrator to load and unload drivers and modules and to query the load status of PSE and its dependents.

By default, the **strload** command loads PSE according to the **/etc/pse.conf** file. The **-f** flag allows the administrator to use an alternate configuration file. The **-d** and **-m** flags are used to specify drivers and modules that are not present in the configuration files (such as when new drivers are being developed). The **-q** flag reports on the system load status (kernel existence) of the referenced drivers and modules.

Configuration File

The configuration file is a flat ASCII, line-oriented database. Comments are introduced by a # (pound sign), and continue until the end of the line. Blank lines are ignored. The form for each record is:

```
attributes filename [argument [node [minor ...] ] ]
```

Fields are separated by spaces, tabs, or both. A - (dash) can be specified as the field value, indicating that the default value is to be used. The fields are defined as follows:

`attributes` Describes the extension to load. The acceptable values are:

- **d** Specifies a driver.
- **m** Specifies a module.
- **s** Creates the node as a standard (not cloned) device.
- **+** Specifies that the extension can be configured more than once. This value must be specified for all lines containing the extension file name.

`filename` Specifies the object file containing the extension. If the command is issued with a ″/″ (slash) in the filename of the driver or module to be loaded, unloaded or queried, the **strload** command uses the value in the filename field explicitly. If there is no ″/″ in the filename entry, the **strload** command will first look for a copy of the driver or module in the current directory. If the driver or module is not in the current directory, **strload** will look for the driver or module in the **/usr/lib/drivers/pse** directory.

> **Note:** It is recommended that the **strload** command be issued from the root directory (/). The **strload** command for load, unload, and query should always be issued from the same directory.

The kernel extension loader REQUIRES that the pathnames used be identical in load, unload and queries. This, coupled with the way the filename is determined by **strload**, could cause problems. Every byte in the pathname used by the **strload** command must EXACTLY match every positionally corresponding byte in the pathname used by the kernel extension loader because the kernel does a **strcmp()** on the filename when looking for matches. If the **strload** command is issued from a different directory to unload the module or driver, one of the following will occur:

- If the **strload** command does not find a copy of the driver or module in the new current directory, **strload** will attempt to unload the driver or module in the **/usr/lib/drivers/pse** directory. However, this pathname may not be the same as the pathname that the loader has logged for that driver or module. If the pathname is not the same, the **strload** command will fail.

- If the **strload** command finds another copy of the module or driver in the new current directory, then the pathnames will be the same, and the loader will correctly unload the driver or module that was loaded. Thus, the **strload** command succeeds, but the results may not be as the user intended.

For example:

The following scenario (NOT recommended) will cause ″spx″, also known as ″A″, to be unloaded. This is probably not the desired effect.

```
mkdir /tmp/foo /tmp/bar
cp /usr/lib/drivers/pse/spx /tmp/foo/A
cp /bin/ls /tmp/bar/A
cd /tmp/foo
strload -d A      # The loader knows the path and filename as
                  # "A" because "A" is found in the current
                  # directory
cd /tmp/bar
strload -q -d A   # Reports "yes" because there is "A" in the
                  # current directory. Note that the file "A"
                  # in /tmp/bar is NOT the same file "A" in
                  # /tmp/foo, but the loader does not care
                  # because it identifies the file by
                  # pathname.
strload -u -d A   # Unloads spx (also known as "A")!
```

The following is an error scenario:

```
mkdir /tmp/foo2 /tmp/bar2
cp /usr/lib/drivers/pse/spx /tmp/foo2/A
cd /tmp/foo2
strload -d A     # The loader knows the path and filename as
                 # "A"because "A" is found in the current
                 # directory.
cd /tmp/bar2
strload -q -d A  # Answers "no". There is no filename
                 # in /tmp/bar2 that matches "A", so strload
                 # prepends pathname "/usr/lib/drivers/pse" to
                 # "A". "/usr/lib/drivers/pse/A" is not found,
                 # so strload answers "no".
strload -u -d A  # Fails - "A" does not exist.
```

The following is an error scenario:

```
cd /usr/lib/drivers/pse
strload -d spx   # The loader knows the path and filename as
                 # "spx" because "spx" is found in the
                 # current directory.
cd /
strload -q -d spx # Answers "no". There is no filename in /
                  # that matches "spx", so strload prepends
                  # the pathname "/usr/lib/drivers/pse" to
                  # "spx". "/usr/lib/drivers/pse/spx" is found
                  # since it exists, so strload gives
                  # "/usr/lib/drivers/pse/spx" to the loader.
                  # The strcmp() fails since
                  # "/usr/lib/drivers/pse/spx" and "spx" do
                  # not match exactly.
strload -u -d spx # Fails - "spx" does not exist.
```

`argument` Has no meaning for the **strload** command. This field is optional. It is passed to the extension when its configuration routine is called. Its interpretation is specific to that extension. The default argument is the value of the `filename` field.

`node` Specifies the name of the node to create. This field is optional. It applies only to drivers and is used as the created node name when the driver is loaded. By default, the created node is **/dev/**`filename`.

`minor` Specifies additional, non clone nodes to create for this driver. This field is optional. The node names are created by appending the minor number to the cloned driver node name. No more than five minor numbers can be given (from 0 to 4), and a node is created for each one.

The **-d** and **-m** flags cause the configuration file to be ignored, unless it is explicitly named on the command line, as follows:

```
strload -f /tmp/my.conf -d newdriver
```

> **Note:** The **-d** and **-m** flags do not override the configuration file. That is, if driver **dgb** is loaded by using the configuration file, the **-d** flag will attempt to reload it but will fail. The configuration file is processed before the **-d** and **-m** flags.

The *List* variable for the **-d** and **-m** flags is a comma-separated list of file names, each of which contains a single PSE driver or module. The configuration process proceeds as if a line of one of the following forms was found in the configuration file:

```
d filename
m filename
```

Flags

-d *List* Lists PSE device drivers to load or unload. The *List* variable specifies a comma-separated list of driver object names.

-f *File* Configures PSE according to the configuration information contained in the file indicated by the *File* variable. The default configuration file is **/etc/pse.conf**.

-m *List* Lists PSE modules to load or unload. The *List* variable specifies a comma-separated list of module object names.

-q Reports load status of extensions.

-u Unloads extensions.

Examples

1. Entering the following command loads PSE (if not already loaded), the `dgb` and `ssb` drivers from the **/usr/lib/drivers/pse/** directory, and the `aoot` module from the current directory, but does not use the configuration file:

   ```
   root# strload -d dgb,ssb -m ./aoot
   ```

2. To unload the `aoot` module only, enter:

   ```
   root# strload -u -m ./aoot
   ```

3. Entering the following command asks if the `spx` driver exists:

   ```
   root# strload -q -d
   spx
   ```

   and produces the following output if not:

   ```
   spx
   : no
   ```

4. The following is an example configuration file:

   ```
   #example configuration file
   d       dgb                                     #line 1
   d       mux    -       -       0                #line 2
   ds      foo                                     #line 3
   d+      xtiso  tcp     /dev/xti/tcp             #line 4
   d+      xtiso  udp     /dev/xti/udp             #line 5
   m       aoot                                    #line 6
   ```

Line 1 loads the dgb driver extension as a cloned device named /dev/dgb. The argument passed to the dgb configuration routine is dgb.

Line 2 loads the mux driver extension as a cloned device named /dev/mux and also creates a standard device name /dev/mux0 with a minor number of 0 (zero). (No more than five device names can be created with minor numbers from 0 to 4.)

Line 3 loads the foo driver extension as a standard device (not cloned) named /dev/foo. The minor number is 0.

Lines 4 and 5 load the xtiso driver extension, and configure it twice: once as tcp and once as udp. The clone nodes created are /dev/xti/tcp and /dev/xti/udp. The configuration routine of xtiso is called twice: once with the argument tcp, and once with udp.

Line 6 loads the aoot module extension. No node is created, and the configuration routine is passed the value aoot.

Files

/usr/lib/drivers/pse/*	Contains PSE kernel extensions.
/etc/pse.conf	Default PSE configuration file.
/usr/sbin/strload	Contains the **strload** command.

Related Information

The **slibclean** command, **strerr** command.

Configuring Drivers and Modules in the Portable Streams Environment (PSE), List of Streams Commands, STREAMS Overview in *AIX 5L Version 5.3 Communications Programming Concepts*.

strreset Command

Purpose

Resets a stream.

Syntax

strreset [**-M** *Major*] [**-m** *Minor*]

Description

The **strreset** command resets an open stream by generating an M_FLUSH message to the stream head. You use it mainly to reset blocked streams. When it is impossible to reopen the stream, issue an I_FLUSH ioctl(), or equivalent command. This situation may happen with a process sleeping in a module's close routine, when signals can not be sent to the process (a zombie process exiting, for example).

Flags

-M *Major*	Specifies the major number for the special file associated with the stream to be reset.
-m *Minor*	Specifies the minor number for the special file associated with the stream to be reset.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Files

/usr/sbin/strreset	Contains the **strreset** command.

strtune Command

Purpose

This command has several related functions:

- Get or set the streams tunable parameters.
- Define the objects to trace using the component trace.
- List the tunable values of the stream modules.
- List the tunable values of the active queues.

Syntax

strtune {**-n** *name* | **-q** *addr*} **-o** *tunable_name*[=*value*] **-o** *tunable_name*[=*value*] ...

strtune [**-n** *name* | **-q** *addr* [**-a**]] **-o** *trclevel*[=*value*]

strtune [**-M**]

strtune [**-Q**]

strtune [**-f** *tunefile*]

Description

There are no restrictions on the use of this command when it is used to display or list values, but when using this command to modify tunable values or to define objects to trace, you must have root authority.

Flags

-n *name*	Defines a stream module name or a device name.
-q *addr*	Defines an active queue address. If the command sets tunables, it modifies the queue pair, or the only queue, depending on the synchronization level of the queue. If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.
-o *tunable_name*	Defines the name of the tunable parameter. Possible values are: • **hiwat** , which defines the high water mark for the flow control on a queue. • **lowat** , which defines the low water mark for the flow control on a queue. • **minpsz** , which defines the minimum packet size. • **maxpsz** , which defines the maximum packet size. A value of -1 indicates an infinite packet size. The **strtune** command can initialize several tunables by listing the **-o** option several times.

value	If no new value is given, the command displays the value of the tunable. Only a user with root authority can modify a tunable parameter value.
-n *name*	Defines a stream module name. If the **-n** or **-q** flag is not present in the command, the command will display or modify the global variable containing the pse global trace level (pse_trclevel).
-q *addr*	Defines an active queue address. If the **-n** or **-q** flag is not present in the command, the command will display or modify the global variable containing the pse global trace level (pse_trclevel). If the command sets the trace level, it modifies the queue pair, or the only queue, depending on the synchronization level of the queue. If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.
-o *trclevel*	Displays or modifies the trace level. The **-o** flag cannot be listed more than once.
value	If no new value is given, the command displays the value of the tunable. Only a user with root authority can modify a tunable parameter value.
-a	Use this flag to force the **strtune** command to propagate the new value to all queues in the stream (from stream head to driver). If the synchronization level is not SQLVL_QUEUE, the synchronization level is also propagated to all queue pairs.
-M	Displays the name, idname, and associated tunable parameter (**minpsz**, **maxpsz**, **lowat**, **hiwat**, **trclevel**) values for each module.
-Q	Displays the name, idname, and associated tunable parameter (**minpsz**, **maxpsz**, **lowat**, **hiwat**, **trclevel**) values for each active queue.
-f *tunefile*	The *tunefile* variable holds the filepath to the file that contains the tunable parameter settings. Each line of the *tunefile* file is managed as one command; if there are any modification commands in the *tunefile*, the user must have root authority for those modifications to be implemented.

Exit Status

0	Successful completion.
>0	An error occurred.

Examples

1. To display the **hiwat** tunable value of **ldterm** module:

   ```
   strtune -n ldterm -o hiwat
   ```

2. With root authority, to set the value of **hiwat** for the **ldterm** module to 8192:

   ```
   strtune -n ldterm -o hiwat=8192
   ```

3. To run the following lines:

   ```
   -n udp6 -o lowat=256
   -n dlpi -o hiwat=4096 -o lowat=128 -o minpsz=128
   ```

   that are listed in the **/tmp/ff** file:

   ```
   strtune -f /tmp/ff
   ```

   This will result in the following commands being run:

   ```
   strtune -n udp6 -o lowat=256
   strtune -n dlpi -o hiwat=4096 -o lowat=128 -o minpsz=128
   ```

File

src/bos/usr/sbin/strtune/strtune.c	Contains the **strtune** command.

Related Information

struct Command

Purpose

Translates a FORTRAN program into a RATFOR program.

Syntax

struct [**-s**] [**-i**] [**-a**] [**-b**] [**-n**] [**-t***Number*] [**-c***Number*] [**-e***Number*] [*File*]

Description

The **struct** command translates the FORTRAN program specified by *File* (standard input default) into a RATFOR program. Wherever possible, RATFOR control constructs replace the original FORTRAN. Statement numbers appear only where still necessary. Cosmetic changes are made, including changing Hollerith strings into quoted strings and relational operators into symbols (for example, **.GT.** into **>**). The output is appropriately indented.

The **struct** command knows FORTRAN 66 syntax, but not full FORTRAN 77. If an input FORTRAN program contains identifiers that are reserved words in RATFOR, the structured version of the program will not be a valid RATFOR program. The labels generated cannot go above 32767. If you get a **goto** statement without a target, try using the **-e** flag.

Flags

-a Turn sequences of **else-if** statements into a non-RATFOR switch of the form:

```
switch
{   case pred1: code
    case pred2: code
    case pred3: code
    default: code
         }
```

The **case** predicates are tested in order. The code appropriate to only one case is executed. This generalized form of **switch** statement does not occur in RATFOR.

-b Generates **goto** statements instead of multilevel **break** statements.

-c *Number* Increments successive labels in the output program by the nonzero integer *Number*. The default is 1. Do not insert a space between **-c** and *Number.*

-e *Number* If *Number* is 0 (default), places code within a loop only if it can lead to an iteration of the loop. Do not insert a space between **-e** and *Number.*

-i Do not turn computed **goto** statements into switches. (RATFOR does not turn switches back into computed **goto** statements.)

-n Generates **goto** statements instead of multilevel **next** statements.

-s Input is accepted in standard format. Comments are specified by a **c**, **C**, or ***** in column 1, and continuation lines are specified by a nonzero, nonblank character in column 6. Input is in the form accepted by the **f77** command.

-t *Number* Makes the nonzero integer *Number* the lowest valued label in the output program. The default is 10. Do not insert a space between **-t** and *Number.*

If *Number* is nonzero, admits small code segments to a loop if otherwise the loop would have exits to several places including the segment, and the segment can be reached only from the loop. In this case, small is close to, but not equal to, the number of statements in the code segment. Values of *Number* under 10 are suggested.

Examples

To translate the `test.f` FORTRAN program into the `newtest.ratfor` RATFOR program, enter:

```
struct -s -i -n -t2 test.f > newtest.ratfor
```

Files

/tmp/struct*	Temporary files used during processing of the **struct** command.
/usr/lib/struct/structure	File that handles processing for the **struct** command.
/usr/lib/struct/beautify	File that handles processing for the **struct** command.
/usr/ucb/struct	Contains the **struct** command.

Related Information

The **asa** or **fpr** command, **fsplit** command.

The Commands overview in *Operating system and device management*.

---

sttinet Method

Purpose

Enables the inet instance.

Syntax

sttinet [**-l** *Interface* ...]

Description

The **sttinet** method enables the inet instance by calling the **ifconfig** command and sets the status flag of the inet instance to AVAILABLE.

Note: The **sttinet** method is a programming tool and should not be executed from the command line.

Flags

-l *Interface ...*	Specifies which specific interface to enable. If no interfaces are specified, then all configured interfaces are started.

Examples

The following method enables the inet instance:

```
sttinet -l tr0 -l tr1
```

Related Information

The **ifconfig** command, **mkdev** command.

The **odm_run_method** subroutine.

Writing a Device Method in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

TCP/IP network interfaces in *Networks and communication management*.

stty Command

Purpose

Sets, resets, and reports workstation operating parameters.

Syntax

stty [**-a**] [**-g**] [*Options*]

Description

The **stty** command sets certain I/O options for the device that is the current standard input. This command writes output to the device that is the current standard output.

This version of the operating system uses the standard X/Open Portability Guide Issue 4 interface to control the terminals, maintaining a compatibility with POSIX and BSD interfaces. The **stty** command supports both POSIX and BSD compliant options, but the usage of POSIX options is strongly recommended. A list of obsolete BSD options, with the corresponding POSIX options, is also provided.

When you redirect standard input from a tty device by typing:

```
stty -a </dev/ttyx
```

the **stty** command (POSIX) will hang while waiting for the **open()** of that tty until the RS-232 carrier detect signal has been asserted. Exceptions to this rule occur if the **clocal** or **forcedcd** (128-port only) option is set.

Flags

-a Writes the current state of all option settings to standard output.
-g Writes option settings to standard output in a form usable by another **stty** command.

Options

The **stty** command supports following categories of options:

- Control Modes
- Input Modes
- Output Modes
- Local Modes
- Hardware Flow Control Modes
- Control Character Assignments
- Combination Modes
- Window Size

Control Modes

clocal	Assumes a line without modem control.
-clocal	Assumes a line with modem control.
cread	Enables the receiver.
-cread	Disables the receiver.
cstopb	Selects 2 stop bits per character.
-cstopb	Selects 1 stop bit per character.
cs5, **cs6**, **cs7**, **cs8**	Selects character size.

hup, **hupcl** Hangs up dial-up connection on the last close.

-hup, **-hupcl** Does not hang up dial-up connection on the last close.

parenb Enables parity generation and detection.

-parenb Disables parity generation and detection.

parodd Selects odd parity.

-parodd Selects even parity.

0 Hangs up phone line immediately.

speed Sets the workstation input and output speeds to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces. Possible values for *speed* are: **50**, **75**, **110**, **134**, **200**, **300**, **600**, **1200**, **1800**, **2400**, **4800**, **9600**, **19200**, **19.2**, **38400**, **38.4**, **exta**, and **extb**.
Note: **exta**, **19200**, and **19.2** are synonyms; **extb**, **38400**, and **38.4** are synonyms.

ispeed *speed* Sets the workstation input speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.

ospeed *speed* Sets the workstation output speed to the specified *speed* number of bits per second. All speeds are not supported by all hardware interfaces, and all hardware interfaces do not support this option. Possible values for *speed* are the same as for the *speed* option.

Input Modes

brkint Signals INTR on break.

-brkint Does not signal INTR on break.

icrnl Maps CR to NL on input.

-icrnl Does not map CR to NL on input.

ignbrk Ignores BREAK on input.

-ignbrk Does not ignore BREAK on input.

igncr Ignores CR on input.

-igncr Does not ignore CR on input.

ignpar Ignores parity errors.

-ignpar Does not ignore parity errors.

inlcr Maps NL to CR on input.

-inlcr Does not map NL to CR on input.

inpck Enables parity checking.

-inpck Disables parity checking.

istrip Strips input characters to 7 bits.

-istrip Does not strip input characters to 7 bits.

iuclc Maps uppercase alphabetic characters to lowercase.

-iuclc Does not map uppercase alphabetic characters to lowercase.

ixany Allows any character to restart output.

-ixany Allows only the START (the Ctrl-Q key sequence) to restart output.

ixoff Sends START/STOP characters when the input queue is nearly empty/full.

-ixoff Does not send START/STOP characters.

ixon Enables START/STOP output control. Once START/STOP output control has been enabled, you can pause output to the workstation by pressing the Ctrl-S key sequence and resume output by pressing the Ctrl-Q key sequence.

-ixon Disables START/STOP output control.

imaxbel Echoes the BEL character and discards the last input character if input overflows.

-imaxbel Discards all input if input overflows.

parmrk Marks parity errors.

-parmrk Does not mark parity errors.

Output Modes

bs0, **bs1**	Selects style of delay for backspaces (**bs0** siginifes no delay).
cr0, **cr1**, **cr2**, **cr3**	Selects style of delay for CR characters (**cr0** siginifes no delay).
ff0, **ff1**	Selects style of delay for form feeds (**ff0** siginifes no delay).
nl0, **nl1**	Selects style of delay for NL characters (**nl0** siginifes no delay).
ofill	Uses fill characters for delays.
-ofill	Uses timing for delays.
ocrnl	Maps CR characters to NL characters.
-ocrnl	Does not map CR characters to NL characters.
olcuc	Maps lowercase alphabetic characters to uppercase on output.
-olcuc	Does not map lowercase alphabetic characters to uppercase on output.
onlcr	Maps NL characters to CR-NL characters.
-onlcr	Does not map NL characters to CR-NL characters.
onlret	On the terminal, NL performs the CR function.
-onlret	On the terminal, NL does not perform the CR function.
onocr	Does not output CR characters at column zero.
-onocr	Outputs CR characters at column zero.
opost	Processes output.
-opost	Does not process output; that is, ignores all other output options.
ofdel	Uses DEL characters for fill characters.
-ofdel	Uses NUL characters for fill characters.
tab0, **tab1**, **tab2**	Selects style of delay for horizontal tabs (**tab0** siginifes no delay).
tab3	Expands tab character to variable number of spaces.
vt0, **vt1**	Selects style of delay for vertical tabs (**vt0** siginifes no delay).

Local Modes

echo	Echoes every character typed.
-echo	Does not echo characters.
echoctl	Echoes control characters as ^X (Ctrl-X), where X is the character given by adding 100 octal to the code of the control character.
-echoctl	Does not echo control characters as ^X (Ctrl-X).
echoe	Echoes the ERASE character as the ″backspace space backspace″ string. **Note:** This mode does not keep track of column position, so you can get unexpected results when erasing such things as tabs and escape sequences.
-echoe	Does not echo the ERASE character, just backspace.
echok	Echoes a NL character after a KILL character.
-echok	Does not echo a NL character after a KILL character.
echoke	Echoes the KILL character by erasing each character on the output line.
-echoke	Just echoes the KILL character.
echonl	Echoes the NL character.
-echonl	Does not echo the NL character.
echoprt	Echoes erased characters backwards with / (slash) and \ (backslash).
-echoprt	Does not echo erased characters backwards with / (slash) and \ (backslash).
icanon	Enables canonical input (canonical input allows input-line editing with the ERASE and KILL characters). See the discussion about canonical mode input in Line Discipline Module (ldterm) in *AIX 5L Version 5.3 Communications Programming Concepts*.
-icanon	Disables canonical input.
iexten	Specifies that implementation-defined functions shall be recognized from the input data. Recognition of the following control characters requires **iexten** to be set: **eol2, dsusp, reprint, discard, werase, lnext.** The functions associated with these modes also require **iexten** to be set: **imaxbel**, **echoke**, **echoprt**, and **echoctl**.
-iexten	Specifies that implementation-defined functions shall not be recognized from the input data.

isig	Enables the checking of characters against the special control characters INTR, SUSP and QUIT.
-isig	Disables the checking of characters against the special control characters INTR, SUSP and QUIT.
noflsh	Does not clear buffers after INTR, SUSP, or QUIT control characters.
-noflsh	Clears buffers after INTR, SUSP, or QUIT control characters.
pending	Causes any input that is pending after a switch from raw to canonical mode to be re-input the next time a read operation becomes pending or the next time input arrives. Pending is an internal state bit.
-pending	No text is pending.
tostop	Signals **SIGTOU** for background output.
-tostop	Does not signal **SIGTOU** for background output.
xcase	Echoes uppercase characters on input, and displays uppercase characters on output with a preceding \ (backslash).
-xcase	Does not echo uppercase characters on input.

Hardware Flow Control Modes

These options are extensions to the X/Open Portability Guide Issue 4 standard.

cdxon	Enables CD hardware flow control mode on output.
-cdxon	Disables CD hardware flow control mode on output.
ctsxon	Enables CTS hardware flow control mode on output.
-ctsxon	Disables CTS hardware flow control mode on output.
dtrxoff	Enables DTR hardware flow control mode on input.
-dtrxoff	Disables DTR hardware flow control mode on input.
rtsxoff	Enables RTS hardware flow control mode on input.
-rtsxoff	Disables RTS hardware flow control mode on input.

Control Assignments

To assign a control character to a character string, type:

```
stty Control String
```

where the *Control* parameter may be the INTR, QUIT, ERASE, KILL, EOF, EOL, EOL2, START, STOP, SUSP, DSUSP, REPRINT, DISCARD, WERASE, LNEXT, MIN, or TIME character. (Use the MIN and TIME characters with the **-icanon** option.)

Note: The values for MIN and TIME are interpreted as integer values, not as character values.

The *String* parameter may be any single character such as `c`. An example of this control assignment is:

```
stty STOP c
```

Another way of assigning control characters is to enter a character sequence composed of a \^ (backslash, caret) followed by a single character. If the single character after the ^ (caret) is one of the characters listed in the ^c (caret c) column of the following table, the corresponding control character value will be set. For example, to assign the DEL control character by using the ? (question mark) character, type the string \^? (backslash, caret, question mark), as in:

```
stty ERASE \^?
```

caret Control Characters in stty

^c	Value
a, A	<SOH>
b, B	<STX>

caret Control Characters in stty

^c	Value
c, C	<ETX>
d, D	<EOT>
e, E	<ENQ>
f, F	<ACK>
g, G	<BEL>
h, H	<BS>
i, I	<HT>
j, J	<LF>
k, K	<VT>
l, L	<FF>
m, M	<CR>
n, N	<SO>
o, O	<SI>
p, P	<DLE>
q, Q	<DC1>
r, R	<DC2>
s, S	<DC3>
t, T	<DC4>
u, U	<NAK>
v, V	<SYN>
w, W	<ETB>
x, X	<CAN>
y, Y	<EM>
z, Z	<SUB>
[	<ESC>
****	<FS>
]	<GS>
^	<RS>
_	<US>
?	<DEL>
@	<NUL>

Combination Modes

cooked See the **-raw** option.

ek Sets ERASE and KILL characters to the Ctrl-H and Ctrl-U key sequences, respectively.

evenp Enables **parenb** and **cs7**.

-evenp Disables **parenb** and sets **cs8**.

lcase, **LCASE** Sets **xcase**, **iuclc**, and **olcuc**. Used for workstations with uppercase characters only.

-lcase, **-LCASE** Sets **-xcase**, **-iuclc**, and **-olcuc**.

nl Sets **-icrnl** and **-onlcr**.

-nl	Sets **icrnl**, **onlcr**, **-inlcr**, **-igncr**, **-ocrnl**, and **-onlret**.
oddp	Enables **parenb**, **cs7**, and **parodd**.
-oddp	Disables **parenb** and sets **cs8**.
parity	See the **evenp** option.
-parity	See the **-evenp** option.
sane	Resets parameters to reasonable values.
raw	Allows raw mode input (no input processing, such as erase, kill, or interrupt); parity bit passed back.
-raw	Allows canonical input mode.
tabs	Preserves tabs.
-tabs, **tab3**	Replaces tabs with spaces when printing.

Window Size

cols *n*, **columns** *n*	The terminal (window) size is recorded as having *n* columns.
rows *n*	The terminal (window) size is recorded as having *n* rows.
size	Prints the terminal (window) sizes to standard output (first rows and then columns).

Obsolete Options

The following BSD options are supported by the **stty** command. For each of them, the recommended POSIX option is given.

all	Use the **stty -a** command to display all current settings.
crt	Use the **sane** option to reset parameters to reasonable values.
crtbs	Use the **-echoe** option.
crterase	Use the **echoe** option.
-crterase	Use the **-echoe** option.
crtkill	Use the **echoke** option.
-crtkill	Use the **echok** and **-echoke** options.
ctlecho	Use the **echoctl** option.
-ctlecho	Use the **-echoctl** option.
decctlq	Use the **-ixany** option.
-decctlq	Use the **ixany** option.
even	Use the **evenp** option.
-even	Use the **-evenp** option.
everything	Use the **stty -a** command to display all current settings.
litout	Use the **-opost** option.
-litout	Use the **opost** option.
odd	Use the **oddp** option.
-odd	Use the **-oddp** option.
pass8	Use the **-istrip** option.
-pass8	Use the **istrip** option.
prterase	Use the **echoprt** option.
speed	Use the **stty** command to display current settings.
tandem	Use the **ixoff** option.
-tandem	Use the **-ixoff** option.

Examples

1. To display a short listing of your workstation configuration, type:

   ```
   stty
   ```

   This lists settings that differ from the defaults.
2. To display a full listing of your workstation configuration, type:

```
stty  -a
```

3. To enable a key sequence that stops listings from scrolling off the screen, type:

   ```
   stty ixon ixany
   ```

   This sets **ixon** mode, which lets you stop runaway listing by pressing the Ctrl-S key sequence. The **ixany** flag allows you to resume the listing by pressing any key. The normal workstation configuration includes the **ixon** and **ixany** flags, which allows you to stop a listing with the Ctrl-S key sequence that only the Ctrl-Q key sequence will restart.

4. To reset the configuration after it has been messed up, type:

   ```
   Ctrl-J stty  sane Ctrl-J
   ```

   Press the Ctrl-J key sequence before and after the command instead of the Enter key. The system usually recognizes the Ctrl-J key sequence when the parameters that control Enter key processing are messed up.

   Sometimes the information displayed on the screen may look strange, or the system will not respond when you press the Enter key. This can happen when you use the **stty** command with parameters that are incompatible or that do things you don't understand. It can also happen when a screen-oriented application ends abnormally and does not have a chance to reset the workstation configuration.

   Entering the **stty sane** command sets a reasonable configuration, but it may differ slightly from your normal configuration.

5. To save and restore the terminal's configuration:

   ```
   OLDCONFIG=`stty -g`          # save configuration
   stty -echo                   # do not display password
   echo "Enter password: \c"
   read PASSWD                  # get the password
   stty $OLDCONFIG              # restore configuration
   ```

   This command saves the workstation's configuration, turns off echoing, reads a password, and restores the original configuration.

   Entering the **stty -echo** command turns off echoing, which means that the password does not appear on the screen when you type it at the keyboard. This action has nothing to do with the **echo** command, which displays a message on the screen.

File

/usr/bin/stty	Contains the **stty** command.

Related Information

The **terminfo** file, **tty** special file **termios.h** header file.

TTY Subsystem Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

ldterm Line Discipline in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

National Language Support in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

stty-cxma Command

Purpose

Sets and reports the terminal options for a TTY configuration of the 128-port asynchronous subsystem.

Syntax

stty-cxma [**-a**] [**-g**] [*Option(s)*] [*ttyName*]

Description

If no flags or options are specified, the **stty-cxma** command reports all 128-port special driver settings and modem signals, as well as all standard parameters reported by the **stty** command for the tty device that is the current standard input.

The *ttyName* parameter can be specified to set or report options for a tty device for other than the standard input. The *ttyName* parameter can be a simple tty name, such as **tty0**, or can be prefixed by **/dev/**, such as **/dev/tty0**. This option may be used on a modem control line when no carrier is present.

Further options can be specified to change flow control settings, set transparent print options, force modem control lines, and display all tty settings. Unrecognized options are passed to the **stty** command for interpretation.

Flags

-a Writes all the unique 128-port settings as well as all the standard tty settings reported by **stty -a** to standard output.

-g Writes option settings to standard output in a form usable by another stty command.

Options

The following options specify transient actions to be performed immediately:

break	Sends a 250 MS break signal out on the tty line.
flush	Discards tty input and output immediately.
flushin	Discards tty input only.
flushout	Discards tty output only.

The actions specified by the following options are in effect until the device is closed. The next time the device is opened, default values are used.

dtr	Raises the DTR modem control line, unless DTR hardware flow control is selected.
-dtr	Drops the DTR modem control line, unless DTR hardware flow control is selected.
rts	Raises the RTS modem control line, unless RTS hardware flow control is selected.
-rts	Drops the RTS modem control line, unless RTS hardware flow control is selected.
startin	Releases flows control to resume stopped input.
startout	Restarts stopped output exactly as if an XON character was received.
stopin	Activates flow control to stop input.
stopout	Stops output exactly as if an XOFF character was received.

2200flow Enables 2200 style flow control on the port. The 2200 terminals support an attached printer and use the following four flow control characters:

0xF8 terminal XON

0xF9 printer XON

0xFA terminal XOFF

0xFB printer XOFF

-2200flow Disables 2200 style flow control on the port.

2200print Runs flow control for the terminal and flow control for the transparent print device (as set by the **2200flow** option) independently.

-2200print Runs terminal and printer flow control (as set by the **2200flow** option) together. So if either the terminal or the printer XOFF character is received, all output is paused until the matching XON character is received.

altpin Switches the location of the DSR and DCD inputs on the modular connector, so that DCD is available when using an 8-pin RJ45 connector instead of the 10-pin RJ45 connector.

-altpin Restores the availability of DSR when using the 10-pin RJ45 connector.

aixon Enables auxiliary flow control, so that two unique characters are used for XON and XOFF. If both XOFF characters are received, transmission will not resume until both XON characters are received.

-aixon Disables auxiliary flow control.

astartc *c* Sets auxiliary XON flow control character. The character may be given as a decimal, octal, or hexadecimal number.

astopc *c* Sets auxiliary XOFF flow control character. The character may be given as a decimal, octal, or hexadecimal number.

bufsize *n* Sets the driver's estimate of the size of the transparent printer's input buffer. After a period of inactivity, the driver bursts this many characters to the transparent printer before reducing to the maximum CPS rate specified by the **maxcps** option rate selected above. The default value is 100 characters.

ctspace Enables CTS hardware output flow control, so local transmission pauses when CTS drops.

-ctspace Disables CTS hardware output flow control.

dcdpace Enables DCD hardware output flow control, so local transmission pauses when DCD drops.

-dcdpace Disables DCD hardware output flow control.

dsrpace Enables DSR hardware output flow control, so local transmission pauses when DSR drops.

-dsrpace Disables DSR hardware output flow control.

dtrpace Enables DTR hardware input flow control, so DTR drops to pause remote transmission.

-dtrpace Disables DTR hardware input flow control.

edelay *n* Sets the rate at which the128-port asynchronous adapter wakes up the driver on input. The adapter wakes the driver every *n* milliseconds. The default value is 100 milliseconds.

fastbaud Alters the baud rate table, so 50 baud becomes 57600 baud.

-fastbaud Restores the baud rate table, so 57500 baud becomes 50 baud.

fastcook Performs cooked output processing on the128-port asynchronous adapter to reduce host CPU usage and increase raw mode input performance.

-fastcook Disables cooked output processing.

forcedcd Disables carrier sense, so the tty may be opened and used even when the carrier is not present.

-forcedcd Reenables carrier sense.

maxchar *n* Sets the maximum number of transparent print characters the driver places in the output queue. Reducing this number increases system overhead; increasing this number delays operator keystroke echo times when the transparent printer is in use. The default value is 50 characters.

maxcps *n* Sets the maximum CPS (characters per second) rate at which characters are output to the transparent print device. The rate chosen should be just below the average print speed. If the number is too low, printer speed is reduced. If the number is too high, the printer resorts to flow control, and user entry on the CRT is impaired accordingly. The default value is 100 CPS.

offstr *s* Sets the CRT escape sequence to turn transparent print off. An arbitrary octal character *xxx* may be given as \xxx.

onstr *s* Sets the CRT escape sequence to turn transparent print on. An arbitrary octal character *xxx* may be given as \xxx.

rtspace Enables RTS hardware input flow control, so RTS drops to pause remote transmission.

-rtspace Disables RTS hardware input flow control.

startc *c* Sets the XON flow control character. The character may be given as a decimal, octal, or hexadecimal number.

stopc *c* Sets the XOFF flow control character. The character may be given as a decimal, octal, or hexadecimal number.

term *t* Sets the transparent printer on and off strings to values specified in the internal default table. Internal defaults are used for the following terminals:**adm31**, **ansi**, **dg200**, **dg210**, **hz1500**, **mc5**, **microterm**, **multiterm**, **pcterm**, **tvi**, **vp-a2**, **vp-60**, **vt52**, **vt100**, **vt220**, **wyse30**, **wyse50**, **wyse60**, or **wyse75**. If the terminal type is not found in the internal default table, the transparent print on and off strings are set to the values specified by the **po** and **pf** attributes in the **termcap** file.

Examples

1. To display all the unique 128-port settings as well as all the standard tty settings for a tty port configured on a 128-port asynchronous controller as /dev/tty0, enter:

   ```
   stty-cxma -a tty0
   ```

2. To make DCD available when using an 8-pin RJ45 connector for a tty port configured on a 128-port asynchronous controller as /dev/tty3, enter:

   ```
   stty-cxma altpin tty3
   ```

   This command interchanges the location of the DSR and DCD inputs on the modular connector.

Files

/usr/ebin/tty/stty-cxma Contains the **stty-cxma** command.

Related Information

The **stty** command.

---

style Command

Purpose

Analyzes surface characteristics of a document.

Syntax

style [**-a**] [**-e**] [**-l***Number*] [**-ml**] [**-mm**] [**-p**] [**-P**] [**-r***Number*] *File* ...

Description

The **style** command analyzes the surface characteristics of the writing style of an English-language document. It reports on readability, sentence length and structure, word length and usage, verb type, and sentence openers. Because the **style** command runs the **deroff** command before looking at the text, header files that contain appropriate formatting information should be included as part of the input.

Note: The use of nonstandard formatting macros may cause incorrect sentence breaks.

Flags

-a	Prints all sentences with their length and readability index.
-e	Prints all sentences that begin with an expletive such as "There are".
-l*Number*	Prints all sentences longer than the number of words specified by the parameter *Number*.
-ml	Causes the **deroff** command to skip lists; use **-ml** if a document contains many lists of sentence fragments.
-mm	Overrides the default **ms** macro package.
-p	Prints all sentences that contain a passive verb.
-P	Prints parts of speech of the words in the document.
-r*Number*	Prints all sentences whose readability index is greater than *Number*.

Related Information

The **diction** command, **deroff** command.

The **ms** macro package.

---

su Command

Purpose

Changes the user ID associated with a session.

Syntax

su [**-**] [*Name* [*Argument ...*]]

Description

The **su** command changes user credentials to those of the root user or to the user specified by the *Name* parameter, and then initiates a new session. The user name may include a DCE cell specification.

Note: The root user is not required to satisfy the Distributed Computing Environment (DCE) authentication when switching to a DCE user. In this case, the user's DCE credentials are not gained.

Any arguments, such as flags or parameters, that are specified by the *Arguments* parameter must relate to the login shell defined for the user specified by the *Name* parameter. These arguments are passed to the specified user's login shell. For example, if the login shell for user `Fred` is **/usr/bin/csh**, you can include any of the flags for the **csh** command, such as the **-f** flag. When the **su** command runs, it passes the **-f** flag to the **csh** command. When the **csh** command runs, the **-f** flag omits the **.cshrc** startup script.

The following functions are performed by the **su** command:

account checking	Validates the user account to be certain it exists, that it is enabled for the **su** command, that the current user is in a group permitted to switch to this account with the **su** command, and that it can be used from the current controlling terminal.
user authentication	Validates the user's identity, using the system-defined primary authentication methods for the user. If a password has expired, the user must supply a new password.
credentials establishment	Establishes initial user credentials, using the values in the user database. These credentials define the user's access rights and accountability on the system.

session initiation	If the **-** flag is specified, the **su** command initializes the user environment from the values in the user database and the **/etc/environment** file. When the **-** flag is not used, the **su** command does not change the directory.

These functions are performed in the sequence shown. If one function is unsuccessful, the succeeding functions are not done. Refer to the **ckuseracct**, **ckuserID**, **authenticate**, **setpcred**, and **setpenv** subroutines for the semantics of these functions.

To restore the previous session, type `exit` or press the Ctrl-D key sequence. This action ends the shell called by the **su** command and returns you to the previous shell, user ID, and environment.

If the **su** command is run from the **/usr/bin/tsh** shell, the trusted shell, you exit from that shell. The **su** command does not change the security characteristics of the controlling terminal.

Each time the **su** command is executed, an entry is made in the **/var/adm/sulog** file. The **/var/adm/sulog** file records the following information: date, time, system name, and login name. The **/var/adm/sulog** file also records whether or not the login attempt was successful: a + (plus sign) indicates a successful login, and a - (minus sign) indicates an unsuccessful login.

Note: Successful use of the **su** command resets the **unsuccessful_login_count** attribute in the **/etc/security/lastlog** file only if the user's **rlogin** and **login** attributes are both set to **false** in **/etc/security/user**. Otherwise, the **su** command doesn't reset the **unsuccessful_login_count**, because the administrator often uses the **su** command to fix user account problems. The user is able to reset the attribute through a local or remote login.

Flags

- Specifies that the process environment is to be set as if the user had logged in to the system using the **login** command. Nothing in the current environment is propagated to the new shell.

Security

The **su** command is a PAM-enabled application with a service name of `su`. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the `usw` stanza of **/etc/security/login.cfg**, to `PAM_AUTH` as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the `su` service in **/etc/pam.conf**. The **su** command requires **/etc/pam.conf** entries for the `auth`, `account`, `password`, and `session` module types. In order for the **su** command to exhibit a similar behavior through PAM authentication as seen in standard AIX authentication, the `pam_allowroot` module must be used as sufficient and called before **pam_aix** in both the `auth` and `account` su service stacks. Listed below is a recommended configuration in **/etc/pam.conf** for the `su` service:

```
#
# AIX su configuration
#
su auth sufficient /usr/lib/security/pam_allowroot
su auth required /usr/lib/security/pam_aix

su account sufficient /usr/lib/security/pam_allowroot
su account required /usr/lib/security/pam_aix

su session required /usr/lib/security/pam_aix

su password required /usr/lib/security/pam_aix
```

Examples

1. To obtain root user authority, type:

   ```
   su
   ```

   This command runs a subshell with the effective user ID and privileges of the root user. You will be asked for the root password. Press End-of-File, Ctrl+D key sequence, to end the subshell and return to your original shell session and privileges.
2. To obtain the privileges of the `jim` user, type:

   ```
   su jim
   ```

   This command runs a subshell with the effective user ID and privileges of `jim`.
3. To set up the environment as if you had logged in as the `jim` user, type:

   ```
   su  - jim
   ```

   This starts a subshell using `jim`'s login environment.
4. To run the backup command with root user authority and then return to your original shell, type:

   ```
   su root "-c /usr/sbin/backup -9 -u"
   ```

   This runs the **backup** command with root user authority within root's default shell. You must give the correct root password when queried for the command to execute.

Files

/usr/bin/su	Contains the **su** command.
/etc/environment	Contains user environment values.
/etc/group	Contains the basic group attributes.
/etc/passwd	Contains the basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/limits	Contains the process resource limits of users.
/etc/security/passwd	Contains password information.
/var/adm/sulog	Contains information about login attempts.

Related Information

The **bsh** command, **csh** command, **getty** command, **ksh** command, **login** command, **setgroups** command, **setsenv** command, **tsh** command, and **tsm** command.

The **authenticate** subroutine, **ckuseracct** subroutine, **ckuserID** subroutine, **setpcred** subroutine, **setpenv** subroutine.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to Securing the network in *Security*.

---

subj Command

Purpose

Generates a list of subjects from a document.

Syntax

subj [*File* ...]

Description

The **subj** command searches one or more English-language files for subjects that might be appropriate in a subject-page index and prints the list of subjects on the standard output. The document should contain formatting commands (from the **nroff**, **troff**, and **mm** commands, among others) to make the best use of the **subj** command.

The **subj** command selects sequences of capitalized words as subjects, except for the first word in each sentence. Thus, if a sentence begins with a proper noun, the capitalization rule does not select this word as a subject. However, since each sentence is expected to begin on a new line, the first word of a sentence that begins in the middle of a line may be erroneously selected. Also, the **subj** command selects modifier-noun sequences from the abstract, headings, and topic sentences (the first sentence in each paragraph). Thus, occasionally a word is incorrectly categorized as a noun or adjective.

The output of the **subj** command may not be appropriate for your needs and should be edited accordingly.

Parameters

File Specifies the English-language files that the **subj** command searches for appropriate subjects for indexing.

Related Information

The **mm** command, **ndx** command, **nroff** command, **troff** command.

---

sum Command

Purpose

Displays the checksum and block count of a file.

Syntax

sum [**-i**] [**-r**] [**-o**] [*File ...*]

Description

The **sum** command reads the file specified by the *File* parameter and calculates a checksum and the number of 1024-byte blocks in that file. If no options are specified, a byte-by-byte algorithm, such as the BSD 4.3 default algorithm, is used. If no files are named, the standard input is read. The checksum and number of 1024-byte blocks are written to standard output. The **sum** command is generally used to determine if a file that has been copied or communicated over transmission lines is an exact copy of the original.

Flags

-i Allows the user to compute the checksum without including header information, if the input file is a binary file. If the input file is not a binary file, the checksum includes header information.

-o Uses the word-by-word algorithm to compute the checksum. The **sum** command with the **-o** flag is compatible with the Version 2 **sum** command in terms of the checksum, but not the number of blocks.

-r Uses a byte-by-byte algorithm to compute the checksum. Using the **-r** flag is the same as using no options.

Note: The default is no longer the word-by-word computation algorithm; it is the BSD 4.3 default algorithm.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

To display the checksum of, and the number of 1024-byte blocks in, the **file1** and **file2** files, type:

```
sum file1 file2
```

If the checksum of the **file1** file is 32830, the checksum of the **file2** file is 32481, and the **file1** file contains one block, and the **file2** contains four blocks, the **sum** command displays:

```
32830           1       file1
32481           4       file2
```

Files

/usr/bin/sum Contains the **sum** command.

Related Information

The **cksum** command, **wc** command.

The File systems in *Operating system and device management* gives an explanation of what a file system is and why to use one.

---

suma Command

Purpose

Creates a task to automate the download of maintenance and technology levels from a fix server.

Syntax

To create, edit, or schedule a SUMA task:

suma { { [**-x**] [**-w**] } | **-s** *CronSched* } [**-a** *Field*=*Value*]... [*TaskID*]

To list SUMA tasks:

suma -l [*TaskID*]...

To list or edit the default SUMA task:

suma -D [**-a** *Field*=*Value*]...

To list or edit the SUMA global configuration settings:

suma -c [**-a** *Field*=*Value*]...

To unschedule a SUMA task:

suma -u *TaskID*

To delete a SUMA task:

suma -d *TaskID*

Description

The **suma** command can be used to perform the following operations on a SUMA task or policy:

- Create
- Edit
- List
- Schedule
- Unschedule
- Delete

The specified operation will be performed on the task represented by a unique Task ID. For the create or edit cases on a SUMA task, if the *TaskID* is not specified, the create operation will be assumed, and a unique *TaskID* will be generated. For the **-l** flag, if *TaskID* is not specified, a list of all SUMA tasks will be displayed. For the **-c** flag, if the **-a** flag is not specified, the SUMA global configuration settings will be listed.

Flags

-c

Lists or edits the SUMA global configuration settings. The **-a** flag allows one or more configuration setting to be updated to the specified value. When used without the **-a** flag, all SUMA configuration settings will be listed.

The configuration settings that can be edited with the **-a** flag are as follows:

FIXSERVER_PROTOCOL
When communicating with the fix server, this specifies whether the transfer will utilize http (non-secure) or https (secure).
Default value: `http`
Allowable values: `http`, `https`

DOWNLOAD_PROTOCOL
When downloading filesets, this specifies whether the transfer will utilize http, ftp, or https (secure) transfers.
Default value: `ftp`
Allowable values: `ftp`, `http`, `https`

DL_TIMEOUT_SEC
Specifies the time in seconds to wait for a response from the fix server during a download operation.
Default value: `180`
Allowable values: Whole numbers greater than zero.

MAX_CONCURRENT_DOWNLOADS
The maximum number of files to be downloaded simultaneously.
Default value: `5`
Allowable values: Whole numbers greater than zero, and ″unlimited″ (or -1).

HTTP_PROXY
Proxy server and port to use for HTTP transfers.
Default value: blank (disabled)
Allowable values:

- Blank (does not use a proxy for HTTP)
- A URL of the following forms:
  - *server.name:port/*; for example, `httpproxy.customer.com:8080/`
  - *protocol://server.name:port/*; for example, `http://httpproxy.customer.com:8080/`
  - *protocol://username:password@server.name:port/*; for example, `http://username:password@httpproxy.customer.com:8080/`

HTTPS_PROXY
Proxy server and port to use for HTTPS transfers.
Default value: blank (disabled)
Allowable values:

- Blank (does not use a proxy for HTTPS)
- A URL of the following forms:
  - *server.name:port/*; for example, `httpsproxy.customer.com:8080/`
  - *protocol://server.name:port/*; for example, `http://httpsproxy.customer.com:8080/`
  - *protocol://username:password@server.name:port/*; for example, `http://username:password@httpsproxy.customer.com:8080/`

-c *(Continued)*

FTP_PROXY
Proxy server and port to use for FTP transfers.
Default value: blank (disabled)
Allowable values:

- Blank (does not use a proxy for FTP)
- A URL of the following forms:
  - *server.name:port/*; for example, `ftpproxy.customer.com:8080/`
  - *protocol://server.name:port/*; for example, `http://ftpproxy.customer.com:8080/`
  - *protocol://username:password@server.name:port/*; for example, `http://username:password@ftpproxy.customer.com:8080/`

SCREEN_VERBOSE
Specifies a verbosity level for logging information to stdout and stderr. Used when the **suma** command is run from the command line or the SMIT interface. It is not applicable for scheduled tasks run from cron.
Default value: `LVL_INFO`
Allowable values:

- `LVL_OFF` : No information is displayed or logged.
- `LVL_ERROR` : Displays error messages and other highly important messages.
- `LVL_WARNING` : Displays warning messages in addition to LVL_ERROR messages.
- `LVL_INFO` : Displays informational messages in addition to LVL_WARNING messages.
- `LVL_VERBOSE` : Displays verbose informational messages in addition to LVL_INFO messages.
- `LVL_DEBUG` : Displays debug output. This setting is for debugging purposes and should not be used for normal operations.

NOTIFY_VERBOSE
Specifies a verbosity level for the information sent in an email notification. Only applies to scheduled tasks run from cron.
Default value: `LVL_INFO`
Allowable values: `LVL_OFF`, `LVL_ERROR`, `LVL_WARNING`, `LVL_INFO`, `LVL_VERBOSE`, `LVL_DEBUG` (refer to the **SCREEN_VERBOSE** setting for value descriptions)

LOGFILE_VERBOSE
Specifies a verbosity level for the information that is logged to the log file (**/var/adm/ras/suma.log**). **Note:** An LVL_OFF setting will still log information to the download log file (**/var/adm/ras/suma_dl.log**).
Default value: `LVL_VERBOSE`
Allowable values: `LVL_OFF`, `LVL_ERROR`, `LVL_WARNING`, `LVL_INFO`, `LVL_VERBOSE`, `LVL_DEBUG` (refer to the **SCREEN_VERBOSE** setting for value descriptions)

MAXLOGSIZE_MB
The maximum size (in MB) that a log file is allowed to reach.
Default value: `1`
Allowable values: Whole numbers greater than zero.

REMOVE_CONFLICTING_UPDATES
Specifies if **lppmgr** should remove conflicting updates that have the same level as base images (**lppmgr -u** flag) when run during a clean action.
Default value: `yes`
Allowable values: `yes`, `no`

REMOVE_DUP_BASE_LEVELS
Specifies whether **lppmgr** should remove duplicate base levels (**lppmgr -b** flag) when run during a clean action.
Default value: `yes`
Allowable values: `yes`, `no`

-c *(Continued)*

REMOVE_SUPERSEDE

Specifies whether **lppmgr** should remove superseded fileset updates (**lppmgr -x** flag) when run during a clean action.
Default value: `yes`
Allowable values: `yes`, `no`

TMPDIR

Specifies the directory to store temporary files.
Default value: **/var/suma/tmp**
Allowable values: Any directory that currently exists.

-d Deletes the SUMA task associated with the given *TaskID* and any schedules for this task that were created with the **-s** flag.

-D Lists or edits the default SUMA task. The **-a** flag allows one or more *Fields* of the default task to be updated to the specified *Value*. When used without the **-a** flag, the default SUMA task will be listed.

-l Lists SUMA tasks. When used without a *TaskID*, all SUMA tasks will be listed. The *TaskID* can be used to specify one or more task IDs to list.

-s *CronSched* Schedules a SUMA task. If specified when a new task is being created, a save is implied (-w flag functionality). The *CronSched* is a list of five space-separated entries (minute, hour, day, month, weekday) contained in quotation marks. The valid values for these entries are as follows (see the **crontab** man page for additional details):

- minute: 0 through 59
- hour: 0 through 23
- day: 1 through 31
- month: 1 through 12
- weekday: 0 through 6 (for Sunday through Saturday)

-u Unschedules a SUMA task. This will remove any scheduling information for the specified *TaskID*.

-w Writes or saves a SUMA task. If used instead of the **-s** flag, the task will be saved, allowing scheduling information to be added later. If used with the **-x** flag, the task will be run immediately and also saved.

-x Specifies that a SUMA task should be run immediately and not scheduled. If used without the **-w** flag, the task will not be saved for future use.

-a *Field*=*Value* ...

Assigns the specified *Value* to the specified *Field*. For the create or edit operation on a SUMA task, the following are the supported *Fields* and *Values*.

RqType

When suma is run with an **RqType** of **Security**, **Critical**, or **Latest**, the **RqType** is the only required field. See example 1 (below) for the default values that will be used in this case. Other **RqType** values (**APAR**, **PTF**, **ML**, **TL**, **SP**, **Fileset**) will require specification of additional *Field*=*Value* information.

APAR Specifies a request to download an APAR. An example is `IY12345`.

PTF Specifies a request to download a PTF. An example is `U123456`.

ML Specifies a request to download a specific maintenance or technology level. An example is `5300-03`.

TL Specifies a request to download a specific maintenance or technology level. An example is `5300-04`.

SP Specifies a request to download a specific service pack or concluding service pack. An example is `5300-04-02` or `5300-04-CSP`.

Fileset Specifies a request to download a specific fileset. An example is `bos.rte.lvm`.

Security
Specifies a request to download the latest security fixes.

Critical Specifies a request to download the latest critical fixes. These resolve highly pervasive (HIPER) problems.

Latest Specifies a request to download all of the latest fixes.

RqName

The specific name of the item requested (for example, `IY12345`, `U123456`, `5200-07`, `5200-08`, `5200-08-02`, `5200-08-CSP`, or `bos.rte.lvm`). The **RqName** field should be left blank when **RqType** equals **Security**, **Critical**, or **Latest**.

RqLevel

The request level can be specified only when the type of item being requested equals **Fileset**, **Latest**, **Security**, **Critical**, or **APAR**. When type equals **Fileset**, this may contain the VRMF of the fileset being requested; for example, `5.3.0.40`. If left blank, the latest level of the fileset is requested.

When type equals **APAR**, **Security**, **Critical**, or **Latest**, an optional maintenance or technology level may be specified (for example `5300–04`) to request fixes on the specified maintenance or technology level without requiring a move to the next maintenance or technology level; for example, `5300–05`.

Repeats

Specifies whether the task will be executed once and not remain on the system, repeat until the item is found, or repeat forever. The **Repeats** field only applies to scheduled tasks run from cron that have an **Action** of **Download** or **Clean**. If run from the command line or if **Action=Preview**, this field is ignored, and no task will be removed.

y Sets up a repeating task, and requires that the task has been assigned a *CronSched* with the **-s** flag. When the **RqType** equals **APAR**, **PTF**, **ML**, **TL**, **SP**, or **Fileset** (with **RqLevel** containing a VRMF), the task will be removed as soon as the item is found. When **RqType** equals **Security**, **Critical**, **Latest**, or **Fileset** (with **RqLevel** containing a blank to imply ″latest″), the task will be set up to repeat forever.

n Specifies the task will be executed once and will not remain on the system.

-a *(Continued)*

DisplayName
Indicates the display name for this SUMA task (for example, ″Download APAR IY12345 when available″). This is used when viewing existing SUMA tasks in SMIT.

Action

Preview
Specifies a download preview will be performed. No filesets will be downloaded.

Download
Specifies that filesets will be downloaded into the **DLTarget** based on the policy.

Clean Specifies that filesets will be downloaded into the **DLTarget** based on the policy, followed by a clean operation. The **lppmgr** command will be used to clean filesets that are not needed from the **DLTarget**. The three configurable **lppmgr** flag options listed in the SUMA global configuration settings are:

- REMOVE_CONFLICTING_UPDATES
- REMOVE_DUP_BASE_LEVELS
- REMOVE_SUPERSEDE

PreCoreqs
Indicates whether prerequisite and corequisite filesets should be downloaded (`y` or `n`).

Ifreqs Indicates whether ifrequisite filesets should be downloaded (`y` or `n`).

Supersedes
Indicates whether filesets that supersede requested filesets should be downloaded instead of the requested filesets (`y` or `n`).

ResolvePE
Indicates what action to take for fixes that have been designated as regressions or PEs. One of three valid values should be specified:

Never Specifies to never resolve regressions and always download specifically what is requested.

Always
Specifies to always resolve regressions. If a resolving fileset is available, it will be downloaded. Otherwise, nothing will be downloaded and this will become a download preview action.

IfAvailable
Specifies to resolve regressions if a resolving fileset is available. If available, the resolving fileset will be downloaded in place of the regression. Otherwise, the fileset designated as a regression will be downloaded.

DLTarget
Contains the directory location where the downloaded files will be stored. If this field is not specified, it will be given the value **/usr/sys/inst.images** and the files will be stored in a directory based on the image type; for example **/usr/sys/inst.images/installp/ppc** or **/usr/sys/inst.images/RPMS/ppc**.

NotifyEmail
Contains one or more e-mail addresses (multiple addresses should be comma-separated) that will be sent a notification e-mail after a fileset download or preview. A notification will only be sent if the task is scheduled for execution at a future time (*CronSched* has been specified).

-a *(Continued)*

FilterDir
Specifies the name of a fix repository directory to filter against so that duplicate fixes will not be downloaded. This allows a directory other than the **DLTarget** to be filtered against. For example, you may filter against a NIM lpp_source without having to download into this directory. If left blank, the **DLTarget** will be used.

FilterML
Specifies a maintenance or technology level to filter against; for example, 5300-04. If not specified, the value returned by **oslevel -r** on the local system will be used.

FilterSysFile
Specifies the name of the file containing the list of installed software to filter against (generated with lslpp -Lc) or the name of a system to filter against. If filtering against the local system, ″localhost″ should be used. If no filtering against installed software is desired, /**dev/null** should be used. In a NIM environment, a NIM client can be specified.

MaxDLSize
The maximum allowable amount of data to be downloaded by any single policy execution, in MB. If it is determined that the download will exceed this size, no download will occur. A value of ″unlimited″ or `-1` can be specified to indicate no upper limit on the amount of data to be downloaded.

Extend Specifying `y` will automatically extend the filesystem where the DLTarget resides. If `n` is specified and additional space is required for the download, no download will occur.

MaxFSSize
The maximum allowable size to which the **DLTarget** filesystem can be extended, in MB. If it is determined that the download will exceed this limit, no download will occur. A value of ″unlimited″ or `-1` can be specified to indicate no upper limit on the size of the filesystem (that is, the filesystem can be expanded until physical disk space is exhausted).

Parameters

TaskID Specifies a unique numeric identifier that is associated with a task. This is assigned when a task is created.

Exit Status

0 The command completed successfully.
>0 An error occurred.

Examples

1. To list the SUMA global configuration settings, type the following:

   ```
   suma -c
   ```

   Output similar to the following will be displayed:

   ```
   FIXSERVER_PROTOCOL=http
   DOWNLOAD_PROTOCOL=ftp
   DL_TIMEOUT_SEC=180
   MAX_CONCURRENT_DOWNLOADS=5
   HTTP_PROXY=
   HTTPS_PROXY=
   FTP_PROXY=
   SCREEN_VERBOSE=LVL_INFO
   NOTIFY_VERBOSE=LVL_INFO
   ```

```
LOGFILE_VERBOSE=LVL_VERBOSE
MAXLOGSIZE_MB=1
REMOVE_CONFLICTING_UPDATES=yes
REMOVE_DUP_BASE_LEVELS=yes
REMOVE_SUPERSEDE=yes
TMPDIR=/var/suma/tmp
```

2. To edit the SUMA global configuration setting to change the maximum log file size to 2 MB, type the following:

```
suma -c -a MAXLOGSIZE_MB=2
```

3. To list the SUMA task defaults, type the following:

```
suma -D
```

   Output similar to the following will be displayed:

```
DisplayName=
Action=Download
RqType=Security
RqName=
RqLevel=
PreCoreqs=y
Ifreqs=y
Supersedes=n
ResolvePE=IfAvailable
Repeats=y
DLTarget=/usr/sys/inst.images
NotifyEmail=root
FilterDir=/usr/sys/inst.images
FilterML=
FilterSysFile=localhost
MaxDLSize=-1
Extend=y
MaxFSSize=-1
```

4. To create and schedule a task that will download the latest critical fixes monthly (e.g., on the 15th of every month at 2:30 AM), type the following:

```
suma -s "30 2 15 * *" -a RqType=Critical \
-a DisplayName="Critical fixes - 15th Monthly"
```

   Note: A task ID will be returned for this newly created task. This example assumes some of the SUMA task defaults, as displayed in the **suma -D** example, will be utilized. For example, when the task default of **DLTarget=/usr/sys/inst.images**, the installp images will be downloaded into the **/usr/sys/inst.images/installp/ppc** directory.

5. To view SUMA scheduling information that has been set up by running a **suma -s CronSched** command, type the following:

```
crontab -l root
```

6. To create and schedule a task that will check for a specific APAR once a week (for example, every Thursday at 3 AM), download it when it becomes available, and send e-mail notifications to users on a remote system, type the following:

```
suma -s "0 3 * * 4" -a RqType=APAR -a RqName=IY12345 \
-a NotifyEmail="bob.smith@host2,ann@host2"
```

   Note: For this task to make a weekly check for an APAR, the **Repeats** field would need to be set to **y**. In this case, after the APAR was found, the task would be deleted. If **Repeats=n**, only a single check would occur before deleting the task.

7. To create and schedule a task that will check for the latest level of the **bos.rte.install** fileset monthly (for example, on the 20th of every month at 4:30 AM), type the following:

```
suma -s "30 4 20 * *" -a RqType=Fileset -a RqName=bos.rte.install \
-a RqLevel=latest -a Repeats=y
```

Note: By setting **Repeats=y**, this task will 'repeat forever' and will not be deleted after a successful download.

8. To create and schedule a task that will check for a specific level of the **bos.rte.install** fileset weekly (for example, every Thursday at 3 AM), type the following:

```
suma -s "0 3 * * 4" -a RqType=Fileset -a RqName=bos.rte.install \
-a RqLevel=5.2.0.30 -a Repeats=y
```

Note: By setting **Repeats=y**, this task will 'repeat until found' and will be deleted after a successful download of the specific fileset.

9. To create and schedule a task that will download the entire AIX 5L Version 5.3 with the 5300-02 Recommended Maintenance package into the **/lppsrc/5302** directory on Monday at 11:00 PM, turning off filtering against installed software, and running an lppmgr clean operation after the download to remove any superseded updates, duplicates base levels, and conflicting updates, type the following:

```
suma -s "0 23 * * 1" -a Action=Clean -a RqType=ML -a RqName=5300-02 \
-a DLTarget=/lppsrc/5302 -a FilterSysFile=/dev/null
```

Note: Prior to running a task which specifies **Action=Clean**, you can run **suma -c** to verify the SUMA global configuration settings that will be used when running lppmgr. In this case, having REMOVE_SUPERSEDE, REMOVE_DUP_BASE_LEVELS, and REMOVE_CONFLICTING_UPDATES all set to **yes** will result in the action described above.

10. To create and schedule a task that will download the entire AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package into the **/tmp/lppsrc/5303** directory on Monday at 11:00 PM, turning off filtering against installed software (allows updates already installed on the system to be downloaded), but filtering against any updates already contained in **/lppsrc**, type the following:

```
suma -s "0 23 * * 1" -a RqType=ML -a RqName=5300-03 \
-a DLTarget=/tmp/lppsrc/5303 -a FilterDir=/lppsrc -a FilterSysFile=/dev/null
```

Note: After the task is successfully completed, the task will be removed, because **RqType=ML** is a 'repeat until found' task. However, if **Repeats=n** only a single check for the 5300-01 ML will be made, and if the ML is not found on the fix server, the task will be deleted because it has been set up not to repeat.

11. To execute a task immediately that will download any critical fixes that have become available since 5300-00 and that are not already installed on your system, type the following:

```
suma -x -a RqType=Critical -a FilterML=5300-00 -a FilterSysFile=localhost
```

Note: This example assumes some of the SUMA task defaults, as displayed in the **suma -D** example, will be utilized. For example, when the task default of **DLTarget=/usr/sys/inst.images**, the installp images will be downloaded into the **/usr/sys/inst.images/installp/ppc** directory.

12. To immediately execute a task that will perform a preview to check if an APAR exists on the fix server, and to create and save this task for later scheduling if the APAR does not yet exist, type the following:

```
suma -x -w -a Action=Preview -a RqType=APAR -a RqName=IY12345
```

Note: A task ID will be returned for this newly created task.

13. To immediately execute the newly created task from the above example (assume task ID 23 was returned) and attempt to download the APAR and save the **Action=Download** setting for task ID 23, type the following:

```
suma -x -w -a Action=Download 23
```

Note: Because his task is being run from the command line, and not scheduled through cron, the **Repeats** field will be ignored and this task will not be deleted regardless of whether the APAR is found.

14. To schedule task ID 23 to repeatedly check for a specific APAR once a week (for example., every Thursday at 3 AM), and download it when it becomes available, type the following:

    ```
    suma -s "0 3 * * 4" -a Repeats=y 23
    ```

    Note: This task will be deleted when the APAR is found.
15. To unschedule a task, which will remove its scheduling information from the crontab file in the **/var/spool/cron/crontabs** directory, type the following:

    ```
    suma -u 23
    ```
16. To delete a task, which also will remove its scheduling information if it exists, type the following:

    ```
    suma -d 23
    ```
17. To list multiple SUMA tasks, where 4 and 23 represent task IDs, type the following:

    ```
    suma -l 4 23
    ```
18. To list all SUMA tasks, type the following:

    ```
    suma -l
    ```
19. To create and schedule a task that will check monthly (for example, on the 15th of every month at 2:30 AM) for all the latest new updates, and download any that are not already in the **/tmp/latest** repository, type the following:

    ```
    suma -s "30 2 15 * *" -a RqType=Latest \
    -a DLTarget=/tmp/latest -a FilterDir=/tmp/latest
    ```

    Note: A task ID will be returned for this newly created task.

Location

/usr/suma/bin/suma

Files

/usr/suma/bin/suma	Contains the **suma** command.
/usr/sbin/suma	Link to **/usr/suma/bin/suma**.
/var/adm/ras/suma.log	Contains detailed results from running the **suma** command.
/var/adm/ras/suma_dl.log	Contains a list of files that have been downloaded.
/var/spool/cron/crontabs	Directory that contains the crontab file for scheduling.

Related Information

The crontab command in *AIX 5L Version 5.3 Commands Reference, Volume 1*.

The lppmgr command in *AIX 5L Version 5.3 Commands Reference, Volume 3*.

---

survd Daemon

Purpose

Controls the surveillance daemon.

Syntax

Controls Surveillance Parameters

survd [**-d** *Delay*] [**-h**]

Resets the Surveillance

survd -r

Description

The **survd** command enables a user with root authority to control the surveillance daemon. You can choose the frequency of the signals that the operating system sends to the bring-up microprocessor (BUMP), using the **-d** *Delay* flag. You can also choose the way the BUMP will reboot the system (hardware or software reboot) if it does not receive a signal from the operating system within the given delay. (The **-h** flag stands for hardware reboot required). Finally, you can decide to turn the surveillance off, using the **-r** flag.

The **survd** daemon works only on multiprocessor systems with Micro Channel® I/O for AIX 5.1 and earlier. For IBM systems, this includes the IBM 7012 Model G Series, the IBM 7013 Model J Series, and the IBM 7015 Model R Seriesfor AIX 5.1 and earlier.

> **Attention:** Do not use the **kill** command to stop the surveillance: in this case the BUMP would no longer receive signals from the daemon and would reboot the system.

Flags

-d *Delay*	Specifies the period of the signals that the operating system sends to the BUMP. The *Delay* parameter indicates the period in seconds. The minimal value authorized for the *Delay* parameter is 10 seconds. If this flag is not specified, the frequency will be 60 seconds.
-h	Indicates that the BUMP will execute a hardware reboot if it does not receive any signal from the operating system within the defined delay. If this flag is not specified a software reboot will be executed.
-r	Turns off the surveillance and kills the surveillance daemon. This flag cannot be used with **-d** or **-h** flags, and it has no action if the daemon was not running.

Security

Access Control: Only the root user can run this command.

Examples

1. To set the signal period to 70 seconds, enter:

   ```
   survd -d 70
   ```

   Note that if no signal is received by the BUMP within 70 seconds, the BUMP will execute a software reboot, since the **-h** flag is not specified.
2. To turn the surveillance off, enter:

   ```
   survd -r
   ```

---

suspendvsd Command

Purpose

suspendvsd – Deactivates an available virtual shared disk.

Syntax

suspendvsd {**–a** | *vsd_name...*}

Description

The **suspendvsd** command brings the specified virtual shared disks from the active state to the suspended state. They remain available. Read and write requests which were active while the virtual shared disk was active are suspended and held. Subsequent read and write operations are also held. If the virtual shared disk is in the suspended state, this command leaves it in the suspended state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Suspend a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

–a Specifies that all the virtual shared disks in the active state are to be suspended.

Parameters

vsd_name Specifies a virtual shared disk. If the virtual shared disk is not in the active state, you get an error message.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To bring the virtual shared disk **vsd1vg1n1** from the active state to the suspended state, enter:

```
suspendvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/suspendvsd

Related Information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startvsd**, **stopvsd**, **ucfgvsd**

svmon Command

Purpose

Captures and analyzes a snapshot of virtual memory.

Syntax

Global Report

svmon -G [**-i** *Interval* [*NumIntervals*]] [**-z**]

User Report

svmon -U [*LogName1...LogNameN*] [**-r**] [**-n** | **-s**] [**-w** | **-f -c**] [**-t** *Count*] [**-u** | **-p** | **-g** | **-v**] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-j**] [**-d**] [**-z**] [**-m**]

Command Report

svmon -C *Command1...CommandN* [**-r**] [**-n** | **-s**] [**-w** | **-f** | **-c**] [**-t** *Count*] [**-u** | **-p** | **-g** | **-v**] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-j**] [**-d**] [**-z**] [**-m**] [**-q** [**s** | **L**]]

Workload Management Class Report

svmon -W [*ClassName1...ClassNameN*] [**-e**] [**-r**] [**-n** | **-s**] [**-w** | **-f** | **-c**] [**-t** *Count*] [**-u** | **-p** | **-g** | **-v**] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-d**] [**-z**] [**-m**] [**-q** [**s** | **L**]]

Workload Management Tier Report

svmon -T [*Tier1...TierN*] [**-a** *SupClassName*] [**-x**] [**-e**] [**-r**] [**-u** | **-p** | **-g** | **-v**] [**-n** | **-s**] [**-w** | **-f** | **-c**] [**-q** [**s** | **L**] [**-t** *Count*] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-z**] [**-m**]

Process Report

svmon -P [*PID1... PIDN*] [**-r** [**-n** | **-s**] [**-w** | **-f** | **-c**] [**-t** *Count*] [**-u** | **-p** | **-g** | **-v**] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-j**] [**-z**] [**-m**] [**-q** [**s** | **L**]]

Segment Report

svmon -S [*SID1...SIDN*] [**-r**] [**-n** | **-s**] [**-w** | **-f** | **-c**] [**-t** *Count*] [**-u** | **-p** | **-g** | **-v**] [**-i** *Interval* [*NumIntervals*]] [**-l**] [**-j**] [**-z**] [**-m**] [**-q** [**s** | **L**]]

Detailed Report

svmon -D *SID1..SIDN* [**-b**] [**-i** *Interval* [*NumIntervals*]] [**-z**] [**-q** [**s** | **L**]]

Framed Report

svmon -F [*Frame1..FrameN*] [**-i** *Interval* [*NumIntervals*]] [**-z**] [**-q** [**s** | **L**]]

Description

The **svmon** command displays information about the current state of memory. The displayed information does *not* constitute a true snapshot of memory, because the **svmon** command runs at user level with interrupts enabled.

The segment is a set of pages and is the basic object used to report the memory consumption. So the statistics reported by **svmon** are expressed in terms of pages.

A page is a block of virtual memory while a frame is a block of real memory. Frames have always a size of 4 KB whereas pages may have different sizes. The base page size is 4 KB. All pages inside a segment have the same size.

Values reported by **svmon** are expressed in 4KB units unless they explicitly pertain to a specific page size, in which case they are expressed in pages of this specific size. For example, in the following report, the first section is expressed in 4KB units. The second and third sections are expressed in pages of **PageSize** and **Psize**, respectively.

```
# svmon -P 1

------------------------------------------------------------------------------
     Pid Command          Inuse      Pin     Pgsp  Virtual 64-bit Mthrd  16MB
 1486924 ptxtstshmlp      14161     7876        0    14160      N     N     Y

     PageSize      Inuse        Pin       Pgsp    Virtual
     s   4 KB      10065       3780          0      10064
     L  16 MB          1          1          0          1

    Vsid      Esid Type Description              PSize Inuse   Pin Pgsp Virtual
       0         0 work kernel segment               s   6083  3777    0  6083
    1640         3 work shared memory segment(lgpg_v L      1     1    0     1
                        sid=ffc1fff)
   148aa         d work loader segment               s   3961     0    0  3961
    b685         2 work process private              s     13     3    0    13
   13689         f work shared library data          s      7     0    0     7
    b605         1 pers code,/dev/hd1:4133           s      1     0    -     -
```

The memory consumption is reported using the *inuse*, *free*, *pin*, *virtual* and *paging space* counters.

- The *inuse* counter represents the number of used frames.
- The *free* counter represents the number of free frames from all memory pools.
- The *pin* counter represents the number of pinned frames, that is, frames that cannot be swapped.
- The *virtual* counter represents the number of pages allocated in the system virtual space.
- The *paging space* counter represents the number of pages reserved or used on paging spaces.

For some reports (**-P**, **-U**, **-C**, **-W**, **-T**), the heading lines display the global memory consumption for the entity (It summarizes the memory consumption of the segments used by this entity). On systems that support alternative page sizes, the memory consumption is displayed, followed by the distribution between the different supported page sizes.

```
   PageSize      Inuse    Pin    Pgsp   Virtual
   s  4 KB        6086   2000    1000      6080
   L 16 MB           1      1       0         1
```

The column headings are:

PageSize
The size of the considered pages. A letter is associated with the page size and is displayed before the size. The size is displayed in kilobytes (KB), megabytes (MB), or gigabytes (GB). The letter associated with the page size is displayed as a reminder and is used in other reports to reference a page size.

Inuse Indicates the total number of pages of **PageSize** in real memory in segments that are used by the user.

Pin Indicates the total number of pages of **PageSize** pinned in segments that are used by the user.

Pgsp Indicates the total number of pages of **PageSize** reserved or used on paging space by segments that are used by the user.

Virtual
Indicates the total number of pages of **PageSize** allocated in the process virtual space.

The different values are expressed in the **PageSize** pages unit.

A segment can be used by multiple processes. Each page from such a segment is accounted for in the *inuse, pin*, *virtual* or *pgspace* fields for each process that uses the segment. Therefore, the total of the *inuse*, *pin*, *virtual* and *pgspace* fields over all active processes may exceed the total number of pages in memory or on paging space.

VMM manages *virtual* page counters for statistical purpose only, which means they are not always up-to-date and their values may be less than the corresponding *inuse* counters.

A segment belongs to one of the five following types:

persistent	Segments used to manipulate files and directories
working	Segments used to implement the data areas of processes and shared memory segments
client	Segments used to implement some virtual file systems like Network File System (NFS) and the CD-ROM file system
mapping	Segments used to implement the mapping of files in memory
real memory mapping	Segments used to access the IO space from the virtual address space.

Implementation Specifics
This command is valid only on the PowerPC® platform.

Reports
The svmon command creates nine types of reports:
- **global**
- **user**
- **command**
- **class**
- **tier**
- **process**
- **segment**
- **detailed segment**
- **frame**

Each report type is described here.

Global Report
The global report is printed when the **-G** flag is specified. The column headings in a global report are:

memory
Specifies statistics describing the use of real memory, including:

size Number of real memory frames (size of real memory)

Note: This includes any free frames that have been made unusable by the memory sizing tool, the **rmss** command.

inuse Number of frames containing pages

Note: On a system where a reserved pool is defined (such as the 16 MB page pool), this value includes the frames reserved for any of these reserved pools.

free Number of frames free of all memory pools

pin Number of frames containing pinned pages

Note: On a system where a reserved pool is defined (such as the 16 MB page pool), this value includes the frames reserved for any of these reserved pools.

virtual Number of pages allocated in the system virtual space

stolen Number of frames stolen by **rmss** and marked unusable by the VMM

in use Specifies statistics on the subset of real memory in use, including:

work Number of frames containing working segment pages

pers Number of frames containing persistent segment pages

clnt Number of frames containing client segment pages

pin Specifies statistics on the subset of real memory containing pinned pages, including:

work Number of frames containing working segment pinned pages

pers Number of frames containing persistent segment pinned pages

clnt Number of frames containing client segment pinned pages

pg space
Specifies statistics describing the use of paging space.

size Size of paging space

inuse Number of paging space pages used

PageSize
Displayed only if alternative page sizes (non-4KB) are available on the system. Specifies individual statistics per page size available on the system.

PageSize
Page size for the following statistics

PoolSize
The number of pages in the pool for a page size using reserved pools (such as the 16 MB page pool)

inuse Number of pages of this size that are used

pgsp Number of pages of this size that are allocated in the paging space

pin Number of pinned pages of this size

virtual Number of pages of this size that are allocated in the system virtual space

User Report

The user report is printed when the **-U** flag is specified. The column headings in a user login report are:

User Indicates the user name

Inuse Indicates the total number of pages in real memory in segments that are used by the user.

Pin Indicates the total number of pages pinned in segments that are used by the user.

Pgsp Indicates the total number of pages reserved or used on paging space by segments that are used by the user.

Virtual
Indicates the total number of pages allocated in the process virtual space.

If processes owned by this user use pages of a size other than the base 4KB page size, these statistics are followed by the distribution between the different page sizes.

If the **-d** flag is specified, these statistics are followed by the information about all the processes run by the specified login user name. This information is similar to the one described in **Process Report**.

Then **svmon** command displays information about the segments used by those processes. This set of segments is separated into three categories:

- Segments that are flagged **system** by the virtual memory manager
- Segments that are only used by the set of processes belonging to the user login name
- Segments that are shared between several users

For each category, there is a report on the page consumption for the related segments. If relevant (see above), a report on the use of the different page sizes is added. Then segment specific metrics are listed.

If the **-l** flag is specified, each segment in the last category is followed by the list of process identifiers that use the segment. Beside the process identifier, the login user name which started it, is also displayed. See the **-l** flag description for special segments processing.

If the **-r** flag is specified, each segment is followed by the ranges, within the segment, where pages have been allocated.

Command Report

The command report is printed when the **-C** flag is specified. The column headings in a command report are:

Command
Indicates the command name.

Inuse Indicates the total number of pages in real memory in segments that are used by the command (for all process running the command).

Pin Indicates the total number of pages pinned in segments that are used by the command (for all process running the command).

Pgsp Indicates the total number of pages reserved or used on paging space by segments that are used by the command.

Virtual
Indicates the total number of pages allocated in the virtual space of the command.

If processes running this command use pages of size other than the base 4KB page size, these statistics are followed by the distribution between the different page sizes.

If the **-d** flag is specified, these statistics are followed by the information about all the processes running the specified command. This information is similar to the one described in **Process Report**.

Next **svmon** displays information about the segments used by those processes. This set of segments is separated into three categories:

- Segments flagged **system** by the virtual memory manager
- Segments only used by the set of processes running the command
- Segments shared between several command names

For each category, there is a report on the page consumption for the related segments. If relevant (see above), a report on the use of the different page sizes is added. Then segment specific metrics are listed.

If the **-l** flag is specified, each segment in the last category is followed by the list of process identifiers that use the segment. Beside the process identifier, the command name it runs is also displayed. See the **-l** flag description for special segments processing.

If the **-r** flag is specified, each segment is followed by the ranges, within the segment, where pages have been allocated.

If the **-q** flag is specified, only information related to segments of the requested size are reported.

Workload Management Reports

svmon provides the ability to report workload management related activity with the following 2 types of report:

- Class Report
- Tier Report

These reports are available when the workload manager is running. Otherwise, the message `WLM must be started` displays and no statistics are reported. When the workload manager is running in passive mode, **svmon** displays the following message: `WLM is running in passive mode` before displaying the statistics.

Refer to Workload Management in *Operating system and device management*.

Class Report

The class report prints when the **-W** flag is specified. The column headings in a class report are:

Class or Superclass
Indicates the class or superclass name.

Inuse Indicates the total number of pages in real memory from segments belonging to the class.

Pin Indicates the total number of pages pinned from segments belonging to the class.

Pgsp Indicates the total number of pages reserved or used on paging space by segments belonging to the class.

Virtual
Indicates the total number of pages allocated in the virtual space of the class.

If the class uses pages of a size other than the base 4KB page size, these statistics are followed by the distribution between the different page sizes.

After these statistics are displayed, **svmon** displays information about the segments belonging to the class.

If the **-e** flag is specified, the statistics of the subclasses of the class are displayed and the segments statistics are reported per subclass. In this case, the class parameter must be a superclass name.

If the **-l** flag is specified, each segment is followed by the list of process identifiers which are using it. Besides the process identifier, the workload class which the process belongs to is also displayed. See the **-l** flag description for special segments processing.

If the **-r** flag is specified, each segment is followed by the ranges, within the segment, where pages have been allocated.

If the **-q** flag is specified, only segments of the requested size are reported. In that case, global metrics are only related to these segments.

Tier Report

The tier report prints when the **-T** flag is specified. The column headings in a tier report are:

Tier Indicates the tier number.

Superclass
Optional column heading. Indicates the superclass name when tier applies to a superclass (when the **-a** flag is used).

Inuse Indicates the total number of pages in real memory from segments belonging to the tier.

Pin Indicates the total number of pages pinned from segments belonging to the tier.

Pgsp Indicates the total number of pages reserved or used on paging space by segments belonging to the tier.

Virtual
Indicates the total number of pages allocated in the virtual space of the tier.

If the tier uses pages of a size other than the base 4KB page size, these statistics are followed by the distribution between the different page sizes.

After these statistics are displayed, **svmon** displays information about the classes belonging to the tier.

If the **-e** flag is specified, the statistics of the subclasses of each superclass belonging to the tier, are reported.

If the **-x** flag is specified, **svmon** displays information about the segments belonging to each class.

The **-r** and **-l** flags can only be used in conjunction with **-x**.

If the **-l** flag is specified, each segment is followed by the list of process identifiers which are using it. Besides the process identifier, the tier number and class which the process belongs to are also displayed. See the **-l** flag description for special segments processing.

If the **-r** flag is specified, each segment is followed by the ranges, within the segment, where pages have been allocated.

If the **-q** flag is specified, only segments of the requested size are reported. In that case, global metrics are only related to these segments.

Process Report

The process report is printed when the **-P** flag is specified. The column headings in a process report are:

Pid Indicates the process ID.

Command
Indicates the command the process is running.

Inuse Indicates the total number of pages in real memory in segments that are used by the process.

Pin Indicates the total number of pages pinned in segments that are used by the process.

Pgsp Indicates the total number of pages reserved or used on paging space by segments that are used by the process.

Virtual
Indicates the total number of pages allocated in the process virtual space.

64-bit Indicates if the process is a 64 bit process (Y) or a 32 bit process (N).

Mthrd Indicates if the process is multi-threaded (Y) or not (N).

16MB Indicates the 16MB page status of the process. ″M″ stands for mandatory, ″Y″ means that the process uses or has used 16MB page segments and ″N″ means that the process does not use 16MB pages.

If the process uses pages of a size other than the base 4KB page size, these statistics are followed by the distribution between the different page sizes.

After process information is displayed, **svmon** displays information about all the segments the process uses. Information about segment are described in the paragraph **Segment Report**

If the **-q** flag is specified, only processes using segments of the requested size are reported. Note that all of the segments for these processes are reported. If a list of processes is specified, an error is reported for each process not using segments of the requested size.

Segment Report

The segment report is printed when the **-S** flag is specified. The column headings in a segment report are:

Vsid Indicates the virtual segment ID. Identifies a unique segment in the VMM.

Esid Indicates the effective segment ID. The Esid is only valid when the segment belongs to the address space of the process. When provided, it indicates how the segment is used by the process. If the vsid segment is mapped by several processes but with different esid values, then this field contains '-'. In that case, the exact esid values can be obtained through the **-P** flag applied on each of the process identifiers using the segment. A '-' also displays for segments used to manage open files or multi-threaded structures because these segments are not part of the user address space of the process.

Type Identifies the type of the segment:

- *pers* indicates a persistent segment
- *work* indicates a working segment
- *clnt* indicates a client segment
- *map* indicates a mapped segment
- *rmap* indicates a real memory mapping segment

Description

Gives a textual description of the segment. The content of this column depends on the segment type and usage (see the segment tables).

If the segment is a persistent segment and is not associated with a log, then the device name and i-node number of the associated file are displayed, separated by a colon. The device name and i-node can be translated into a file name with the **ncheck** command or by using the **-j** flag. If the segment is the primary segment of a large file, then the words `large file` are prepended to the description.

Note: Mapping device name and inode number to file names can be a lenghtly operation for deeply nested filesystems. Because of that, the **-j** option should be used with caution.

If the segment is a persistent segment and is associated with a log, then the string `log` displays. If the segment is a working segment, then the **svmon** command attempts to determine the role of the segment. For instance, special working segments such as the kernel and shared library are recognized by the **svmon** command. If the segment is the private data segment for a process, then `private` prints out. If the segment is the code segment for a process, and the segment report prints out in response to the **-P** flag, then the string `code` is prepended to the description.

If the segment is mapped by several processes and used in a different way (for example, a process private segment mapped as shared memory by another process), then the description is empty. The exact description can be obtained through **-P** flag applied on each process identifier using the segment.

If a segment description is too large to fit in the description space, then the description is truncated. The truncated part can be obtained through the **-S** flag (without **-I**) on a given segment.

Segment Tables

Description Table

Segment Type	Segment usage	Description
persistent	log files	log
persistent	files and directories	device name : inode number
persistent	large files	large file device name : inode number
mapping	files mapping	mapped to sid source sid no longer mapped
working	data areas of processes and shared memory segments	dependent on the role of the segment based on the VSID and ESID
client	NFS and CD-ROM files	dependent on the role of the segment based on the VSID and ESID
rmapping	IO space mapping	dependent on the role of the segment based on the VSID and ESID

Description Table Based on vsid for a 32-bit Kernel

Segment Type	Description
segment table	segment table
kernel segment	kernel seg
kernel extension segment	kernel ext seg
software page frame segments	page frame table
ram disk segment	ram disk
user's shadow of kernel segment	kernel shadow
paging space disk map segment	page space disk map
base config segment	base config seg
software hat segment	software hat
file structures segment	misc kernel tables
kernel heap segment	kernel heap
mbuf pool segment	mbuf pool
address map entry segment	ame segment
page table area segment	page table area

Description Table Based on vsid for a 64-bit Kernel

Segment Type	Description
kernel segment	kernel segment
VMMs private segment	vmm data segment
segment table segments	vmm scb segment
page table area segment	page table area
paging space disk map segment	page space disk map
software hat segments	vmm software hat
real memory heap segment	vmm rmalloc heap
address map entry segments	vmm ame segment
vmm vmintervals segment	vmm vmintervals
other vmm segments	other vmm segments
process and thread tables	process and thread tables
mbuf pool segment	mbuf pool
loader segment	loader segment
jfs segment	jfs segment

Segment Type	Description
lfs segment	lfs segment
lock instrumentation segment	lock instrumentation
kernel heap segment	kernel heap
heap debug segment	heap debug
mpdata debug segment	mpdata debug
other kernel segments	other kernel segments

Description Table Based on esid for 32-bit Processes on a 32-bit Kernel

ESID value or ranges	Description
0x0	kernel
0x1	code
0x2	process private
0x3-0xC	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0xD	shared library text
0xE	shared library ovfl
0xF	shared library data

Description Table Based on esid for 64-bit Processes on a 32-bit Kernel

ESID value or ranges	Description
0x0	kernel
0xD	shared library text
0x10 - 0x6FFFFFFF	text data BSS heap
0x70000000 -0 x7FFFFFFF	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0x80000000 - 0x8FFFFFFF	private load

ESID value or ranges	Description
0x90000000 - 0x90010009	shared library text
0x9001000A - 0x90020013	shared library data
0x90020014 - 0x9FFFFFFE	shared library
0x9FFFFFFF	USLA
0xA0000000 - 0xEFFFFFFF	rsvd for system use
0xF0000000 - 0xFFFFFFFF	application stack
0xF00000002	process private

Description Table Based on esid for 32-bit Processes on a 64-bit Kernel

ESID value or ranges	Description
0x0	kernel
0x1	code
0x2	process private
0x3-0xC	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0xD	shared library text
0xE	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0xF	shared library data
0x9FFFFFD0-0x9FFFFFDF	32 bit user alias
0x9FFFFFF0-0x9FFFFFFF	32bit loader alias

Description Table Based on esid for 64-bit Processes on a 64-bit Kernel

ESID value or ranges	Description
0x0	kernel
0x2	process private

ESID value or ranges	Description
0x3-0xC	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0xE	• unused segment • shared memory segment • mapped file • file mapped read write • defered update • mmap segment • working storage • RMMAP region segment • other segments • extended shm segments • text or shared-lib code segment • overflow heap • unknown
0x10-0x6FFFFFFF	text data BSS heap
0x70000000-0x7FFFFFFF	default shmat/mmap
0x80000000-0x8FFFFFFF	private load
0x90000000-0x9FFFFFFF	shared library text
0xF0000000-0xFFFFFFFF	application stack

PSize Indicates the size of the pages inside the segment. The page size is represented by a letter (for example, **s** for 4KB pages and **L** for 16MB pages).

Inuse Indicates the number of pages in real memory in this segment.

Pin Indicates the number of pages pinned in this segment.

Pgsp Indicates the number of pages used on paging space by this segment. This field is relevant only for working segments.

Virtual
Indicates the number of pages allocated for the virtual space of the segment. (Only for working segments).

If the **-r** flag is specified, the ranges within the segment pages which have been allocated. A working segment may have two ranges because pages are allocated by starting from both ends and moving towards the middle.

If the **-l** flag is specified, the list of process identifiers which use that segment is displayed. See the **-l** flag description for special segments processing.

If the **-q** flag is specified, only segments of the requested size are reported. If a list of segments is specified, an error is reported for each segment of a different size..

Note: Segment reports can only be generated for primary segments.

Detailed Report

The detailed report is printed when the **-D** flag is specified.

Several fields are presented before the listing of the used pages:

Segid
The segment identifier.

Type The type of the segment.

PSize Indicates the size of the pages inside the segment.

Address Range
Ranges in which frames are used by this segment.

For working segments, the following additional fields are also present:

Size of page space allocation
Number of pages paged out.

Virtual
Number of pages used by this segment.

Inuse Number of frames used by this segment.

Column headings in a detailed report.

Same segment information as described in the segment report and additionally for each frame:

Page Relative page number to the virtual space. This page number can be higher than the number of frames in a segment (65535) if the virtual space is larger than a single segment (large file).

Frame Frame number in the real memory. Since frames are always considered 4KB in size regardless of the page size of the segment, for any page size larger than 4 KB, a range of frames instead of a single frame is associated to one page. This range is noted as *XXXXXXX*..*YYYYYYY*, which means that the continuous range of frames between ID *XXXXXXX* and *YYYYYYY* is used for the given page.

Pin Indicates if the frame is pinned or not

Ref Indicates if the frame has been referenced by a process (**-b** flag only).

Mod Indicates if the frame has been modified by a process (**-b** flag only).

ExtSegid
Extended segment identifier. This field is only set when the page number is higher than the maximum number of frames in a segment.

ExtPage
Extended page number. This field is only set when the page number is higher than the maximum number of frames in a segment and indicates the page number within the extended segment.

Note: Detailed reports can only be generated for primary segments.

Frame Report

The frame report prints when the **-F** flag is specified.

When no argument is specified, the frame report returns the percentage of usage of reserved pools (such as the 16 MB page pool) if defined. The reported numbers take into account only frames whose reference bit is set. During this processing, all reference bits are reset. As a result, the second time this command is used, it reports the percentage of real memory accessed since the previous time. This helps determine the activity intensity of the machine. For example:

```
# svmon -F -i 10 3
 Processing.. 100%
 percentage of memory used:   73.85%
 percentage of 16MB page memory used:   0.00%
 Processing.. 100%
 percentage of memory used:   12.84%
 percentage of 16MB page memory used:   0.00%
 Processing.. 100%
 percentage of memory used:   11.25%
 percentage of 16MB page memory used:   0.00%
```

Note: This processing interferes with the page stealer (LRU) activity, which also uses the reference bit to determine which pages to page out.

When frame numbers are specified, the column headings in the report are:

Frame Frame number in real memory.

Segid Indicates the virtual segment ID that the frame belongs to (the primary segment ID in case of extended segment).

Ref Indicates if the frame has been referenced by a process.

Mod Indicates if the frame has been modified by a process.

Pincount
Indicates the long term pincount and the short term pincount for the frame.

State Indicates the state of the frame (Bad, In-Use, Free, I/O, PgAhead, Hidden).

Swbits
Indicates the status of the frame in the Software Page Frame Table.

ExtSegid
Extended segment identifier. This field is only set when the frame belongs to an extended segment.

PSize Indicates the size of the page to which the frame belongs. The page size is represented by a letter (for example, **s** for 4KB pages and **L** for 16MB pages).

Note: To generate a Segment or Detailed report for the segment a frame belongs to, always use **Segid**, because those reports cannot be generated for extended segments.

Flags

If no command line flag is given, then the **-G** flag is the default.

-a *SupClassName*	Restricts the scope to the subclasses of the *SupClassName* class parameter (in the Tier report **-T**). The parameter is a superclass name. No list of class is supported.
-b	Shows the status of the reference and modified bits of all the displayed frames (detailed report **-D**). When shown, the reference bit of the frame is reset. When used with the **-i** flag, it detects which frames are accessed between each interval. **Note:** Use this flag with caution because of its impact on performance.
-c	Indicates that only client segments are to be included in the statistics. By default all segments are analyzed.

-C *Command1...CommandN*
Displays memory usage statistics for the processes running the command name *Commandnm1...CommandnmN*. *Commandnm* is a string. It is the exact basename of an executable file.

-d
Displays for a given entity, the memory statistics of the processes belonging to the entity (user name or command name).

-D *SID1...SIDN*
Displays memory-usage statistics for segments *SID1...SIDN*, and a detail status of all frames of each segment. Segment ids specified must be of primary segments.

-e
Displays the memory usage statistics of the subclasses of the *Class* parameter in the Workload Class report **-W** and in the Tier report **-T**. The class parameter of **-W** or **-a** needs to be a superclass name.

-f
Indicates that only persistent segments (files) are to be included in the statistics. By default all segments are analyzed.

-F [*Frame1...FrameN*]
Displays the status of frames *Frame1...FrameN* including the segments to which they belong. If no list of frames is supplied, the percentage of memory used displays.

-g
Indicates that the information to be displayed is sorted in decreasing order by the total number of pages reserved or used on paging space. This flag, in conjunction with the segment report, shifts the non-working segment at the end of the sorted list.

-G
Displays a global report.

-i *Interval* [*NumIntervals*]
Instructs the **svmon** command to display statistics repetitively. Statistics are collected and printed every *Interval* seconds. *NumIntervals* is the number of repetitions; if not specified, **svmon** runs until user interruption, Ctrl-C.
Note: Because it may take a few seconds to collect statistics for some options, the observed interval may be larger than the specified interval.

-j
Shows, for each persistent segment, the file path referred.
Note: This flag should be used with caution because of its potential performance impacts (especially with **svmon -S**).

-l
Shows, for each displayed segment, the list of process identifiers that use the segment and, according to the type of report, the entity name (login, command, tier, or class) to which the process belongs. For special segments, a label is displayed instead of the list of process identifiers.

System segment
This label is displayed for segments that are flagged *system*

Unused segment
This label is displayed for segments which are not used by any existing processes. For example, persistent segments relative to files no longer in usage.

Shared library text
This label is displayed for segments that contain text of shared library, and that may be used by most of the processes (for example, **libc.a**). This is to prevent the display of a long list of processes.

-m
Displays information about source segment and mapping segment when a segment is mapping a source segment. The default is to display only information about the mapping segment.

-n
Indicates that only non-system segments are to be included in the statistics. By default all segments are analyzed.

-p
Indicates that the information to be displayed is sorted in decreasing order by the total number of pages pinned.

-P [*PID1*... *PIDN*]	Displays memory usage statistics for process *PID1*...*PIDN*. *PID* is a decimal value. If no list of process IDs (PIDs) is supplied, memory usage statistics are displayed for all active processes.
-q [**s** | **L**]	Filters results regarding whether they deal with pages of the requested size or not. The requested page size is specified through the option sub-argument. 4KB pages are specified with **s** and 16MB pages with **L**. If no sub-argument is specified, the default is 16MB pages (**L**). Metrics per page size are also displayed.
-r	Displays the ranges within the segment pages which have been allocated. A working segment may have two ranges because pages are allocated by starting from both ends and moving towards the middle.
-s	Indicates that only system segments are to be included in the statistics. By default all segments are analyzed.
-S [*SID1*...*SIDN*]	Displays memory-usage statistics for segments *SID1*...*SIDN*. *SID* is a hexadecimal value. Segment IDs specified must be of primary segments. If no list of segment IDs (SIDs) is supplied, memory usage statistics are displayed for all defined segments.
-t *Count*	Displays memory usage statistics for the top *Count* object to be printed
-T [*Tier1...TierN*]	Displays memory usage statistics of all the classes of the tier numbers *Tier1...TierN*. If no list of tier is supplied, memory usage statistics displayed for all the defined tiers.
-u	Indicates that the information to be displayed is sorted in decreasing order by the total number of pages in real memory. It is the default sorting criteria if none of the following flags are present: **-p**, **-g** and **-v**.
-U [*LogName1*...*LogNameN*]	Displays memory usage statistics for the login name *LogName1*...*LogNameN*. *LogName* is a string, it is an exact login name. If no list of login identifier is supplied, memory usage statistics are displayed for all defined login identifiers.
-v	Indicates that the information to be displayed is sorted in decreasing order by the total number of pages in virtual space. This flag, in conjunction with the segment report, shifts the non-working segment at the end of the sorted list.
-w	Indicates that only working segments are to be included in the statistics. By default all segments are analyzed.
-W [*Clnm1*...*ClnmN*]	Displays memory usage statistics for the workload management class *Clnm1*...*ClnmN*. *Clnm* is a string. It is the exact name of a class. For a subclass, the name should have the form *superclassname.subclassname*. If no list of class name is supplied, memory usage statistics display for all defined class names.
-x	Displays memory usage statistics for the segments for every class of a tier in the Tier report **-T**.
-z	Displays the maximum memory size dynamically allocated by **svmon** during its execution.

Security

Access Control: You must have root authority to run this command.

Examples

1. To print out global statistics, type:

```
svmon -G
```

```
svmon -G
               size      inuse       free        pin    virtual
memory       786432      99980     686452      61545      86822
pg space      16384       1084
```

```
             work       pers       clnt
pin           53353          0          0
in use        70261      21527          0
PageSize   PoolSize      inuse       pgsp        pin    virtual
s   4 KB          -      83596       1084      45161      78630
L  16 MB          4          2          0          4          2
```

The memory size of the system is 786432 frames (786432*4*1024 bytes, or 208 GB). This size is split into the inuse frames (99980 frames) and the free frames (686452 frames).

61545 frames are pinned in memory.

The number of pages allocated in the system virtual space is 86822.

The inuse and pin values include the pages reserved for the 16MB page memory pool.

The size of the paging space is 16384 pages (16384*4*1024 bytes, or 64 MB).

The number of paging space pages used is 1084.

The pinned frames (61545 frames) is composed of working segment pinned pages (53353 frames), not counting the memory not used but pinned by the 16MB page pool.

The number of frames containing pages (99980 frames) is composed of working segment pages (70261 frames) and persistent segment pages (21527 frames), not counting the memory that is only reserved but counted inuse from the 16MB pool.

Then statistics are displayed for each page size available on the system. For instance, the 16MB page pool is composed of 4 pages of 16MB. Two of these are used, none are in the paging space (since they are all pinned), all of these are pinned, and 2 are in the system's virtual space.

The values are expressed in **PageSize** units. The 16MB page pool is equivalent to 16384 pages of 4KB.

2. To print out global statistics each minute during 1/2 hour, type:

```
svmon -G -i 60 30
```

3. To print out the memory statistics for the user *steve*, type:

```
svmon -U steve
```

```
================================================================================
User                                Inuse        Pin       Pgsp    Virtual
steve                               16309      11269       1083      16304
       PageSize       Inuse      Pin        Pgsp       Virtual
       s   4 KB       8117       3077       1083          8112
       L  16 MB          2          2          0             2
................................................................................
SYSTEM segments       Inuse      Pin        Pgsp       Virtual
                      5828       3074       1083          5823
       PageSize       Inuse      Pin        Pgsp       Virtual
       s   4 KB       5828       3074       1083          5823
       L  16 MB          0          0          0             0
   Vsid     Esid Type Description              PSize  Inuse   Pin  Pgsp  Virtual
      0        0 work kernel segment               s  5823    3074 1083     5823
................................................................................
EXCLUSIVE segments              Inuse      Pin       Pgsp    Virtual
                                8686       8195         0       8686
       PageSize      Inuse       Pin        Pgsp   Virtual
       s   4 KB        494         3           0       494
       L  16 MB          2         2           0         2
   Vsid     Esid Type Description              PSize  Inuse   Pin  Pgsp  Virtual
  4d093 70000000 work default shmat/mmap            L      2     2     0        2
  5ceb7        2 work process private               s    494     3     0      494
  44ed1        - pers /dev/hd9var:4228              s      0     0     -        -
  64e79        - pers /dev/hd9var:37                s      0     0     -        -
................................................................................
SHARED segments                 Inuse      Pin       Pgsp    Virtual
                                 1795        0          0       1795
       PageSize      Inuse       Pin        Pgsp   Virtual
       s   4 KB       1795         0          0      1795
```

```
      L  16 MB          0         0           0         0
   Vsid     Esid Type Description              PSize  Inuse   Pin  Pgsp  Virtual
  289aa        d work shared library text           s   1795     0     0     1795
```

The user steve has 16309 pages in real memory. 11269 frames are pinned.

The number of pages reserved or used in paging space is 1083. The number of pages allocated in the virtual space is 16304.

Among the pages used by the user steve, there are 2 of them that pertain to the 16MB page pool.

The segments used by the processes run by the user steve are separated into three categories:

- SYSTEM: segments shared by all processes
- EXCLUSIVE: segments used by the set of processes belonging to steve
- SHARED: segments shared by several users.

The global statistics for the user steve is the sum of each field (**Inuse**, **Pin**, **Pgsp**, **Virtual**) of these three categories (SYSTEM, EXCLUSIVE, SHARED).

4. To print out the memory usage statistics for the users root and steve taking into account only working segments, type:

   ```
   svmon -U root steve -w
   ```

5. To print out the top 10 users of the paging space, type:

   ```
   svmon -U -g -t 10
   ```

6. To print out the memory usage statistics for the user *steve*, including the list of the process identifiers, type:

   ```
   svmon -U steve -l
   ```

```
================================================================================
User                                Inuse        Pin       Pgsp    Virtual
steve                               16309      11269       1083      16244
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       8117       3077       1083       8052
    L  16 MB          2          2          0          2
--------------------------------------------------------------------------------
SYSTEM segments                     Inuse        Pin       Pgsp    Virtual
                                     5828       3074       1083       5823
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       5828       3074       1083       5823
    L  16 MB          0          0          0          0
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
      0         0 work kernel segment               s   5823  3074 1083    5823
--------------------------------------------------------------------------------
EXCLUSIVE segments                  Inuse        Pin       Pgsp    Virtual
                                     8686       8195          0       8686
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB        494          3          0        494
    L  16 MB          2          2          0          2
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
  4d093  70000000 work default shmat/mmap           L      2     2    0       2
  5ceb7         2 work process private              s    494     3    0     494
  44ed1         - pers /dev/hd9var:4228             s      0     0    -       -
  64e79         - pers /dev/hd9var:37               s      0     0    -       -
--------------------------------------------------------------------------------
SHARED segments                     Inuse        Pin       Pgsp    Virtual
                                     1795          0          0       1735
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1795          0          0       1735
    L  16 MB          0          0          0          0
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
  289aa         d work shared library text          s   1735     0    0    1735
   8050         1 pers code,/dev/hd2:4204           s     58     0    -       -
                pid:12374    user: steve
                pid:11194    user: root
                pid:10870    user: root
                pid:10574    user: root
```

```
                pid:10000    user: root
                pid:6544     user: root
   1863         - pers /dev/hd2:12837              s      2     0    -       -
                pid:12374    user: steve
                pid:11194    user: root
                pid:10870    user: root
                pid:10574    user: root
                pid:10000    user: root
                pid:6544     user: root
```

Since the **-I** flag is specified, each segment of the SHARED segments category is followed by the list of process identifiers and user names which are using it.

7. To print out the memory usage statistics for the user *steve* including the processes belonging to *steve,* type:

```
svmon -U steve -d
================================================================================
User                                Inuse        Pin       Pgsp    Virtual
steve                               16309      11269       1083      16304
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       8117       3077       1083       8112
    L  16 MB          2          2          0          2
--------------------------------------------------------------------------------
    Pid Command          Inuse        Pin     Pgsp  Virtual 64-bit Mthrd  16MB
1032304 shm_lgpg64       16309      11269     1083    16304      Y     N     Y
................................................................................
SYSTEM segments                     Inuse        Pin       Pgsp    Virtual
                                     5828       3074       1083       5823
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       5828       3074       1083       5823
    L  16 MB          0          0          0          0
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
      0         0 work kernel segment              s   5823  3074 1083    5823
................................................................................
EXCLUSIVE segments                  Inuse        Pin       Pgsp    Virtual
                                     8686       8195          0       8686
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB        494          3          0        494
    L  16 MB          2          2          0          2
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
  4d093  70000000 work default shmat/mmap           L      2     2    0       2
  5ceb7         2 work process private              s    494     3    0     494
  44ed1         - pers /dev/hd9var:4228             s      0     0    -       -
  64e79         - pers /dev/hd9var:37               s      0     0    -       -
................................................................................
SHARED segments                     Inuse        Pin       Pgsp    Virtual
                                     1795          0          0       1795
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1795          0          0       1795
    L  16 MB          0          0          0          0
   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
  289aa         d work shared library text          s   1795     0    0    1795
```

Since the **-d** flag is specified, the global statistics are followed by the information about all the processes run by steve (in this case, only the **shm_lgpg64** command).

8. To print out the memory usage statistics for the **login** command , including the address ranges, type:

```
svmon -C login -r
================================================================================
Command                             Inuse        Pin       Pgsp    Virtual
login                                7092       3130       1083       6844
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       7092       3130       1083       6844
................................................................................
SYSTEM segments                     Inuse        Pin       Pgsp    Virtual
```

```
                                    5025       3127       1083       5025
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       5025       3127       1083       5025
  Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
     0         0 work kernel segment               s   5025  3127 1083    5025
                 Addr Range: 0..23941
.....................................................................................
EXCLUSIVE segments                  Inuse        Pin       Pgsp    Virtual
                                      103          3          0        103
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB        103          3          0        103
  Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
 3ad9d         2 work process private              s     54     3    0      54
                 Addr Range: 0..35 : 65310..65535
 3cd9e         f work shared library data          s     49     0    0      49
                 Addr Range: 0..1080
.....................................................................................
SHARED segments                     Inuse        Pin       Pgsp    Virtual
                                     1964          0          0       1716
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1964          0          0       1716
  Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
 149aa         d work shared library text          s   1716     0    0    1716
                 Addr Range: 0..2800
 3eb9f         - pers /dev/hd2:202789              s    194     0    -       -
                      Addr Range: 0..947
 10ba8         - pers /dev/hd2:202790              s     36     0    -       -
                      Addr Range: 0..35
   d80         1 pers code,/dev/hd2:297192         s     15     0    -       -
                      Addr Range: 0..14
 24c32         - pers /dev/hd4:8200                s      3     0    -       -
                      Addr Range: 0..2
```

The login command uses 7092 pages in real memory. 3130 frames are pinned.

The number of pages reserved or used in paging space is 1083.

The number of pages allocated in the virtual space is 6844. The login command does not use pages other than 4KB pages.

The segments used by the processes running the login command are separated into three categories:

- SYSTEM: segments shared by all processes
- EXCLUSIVE: segments used by the set of processes running the login command
- SHARED: segments shared by several users

The global statistics for the login command is the sum of each field (Inuse, Pin, Pgsp, Virtual) of these three categories (SYSTEM, EXCLUSIVE, SHARED).

The address range for each segment is displayed after its statistics.

9. To print out memory usage statistics for the commands oracle, xemacs and cc, type:

   ```
   svmon -C oracle xemacs cc
   ```

10. To print out the memory usage statistics for the superclass *System*, type:

    ```
    svmon -W System
    ================================================================================
    Superclass                          Inuse        Pin       Pgsp    Virtual
    System                               2111        465       1128       2394
        Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
        700e         - work kernel pinned heap           s   1235   460 1128    1638
        af90         - work                              s    391     1    0     391
        9f76         - work                              s    223     0    0     214
        a771         - work                              s     67     1    0      67
          80         - pers /dev/hd2:3                   s     61     0    -       -
        cf7c         - work                              s     47     0    0      35
        f79b         - work                              s     21     0    0      14
    ```

```
    9052         - pers /dev/hd3:4                  s     17     0    -       -
    37a3         - work                             s     15     2    0      15
     7a5         - work                             s     15     0    0       8
    9797         - work                             s     12     1    0      12
    52aa         - pers /dev/hd2:4196               s      3     0    -       -
    786f         - pers log                         s      2     0    -       -
    ca1a         - pers /dev/hd2:12817              s      1     0    -       -
    89b1         - pers /dev/hd4:761                s      1     0    -       -
```

The class System uses 2111 pages in real memory. 465 frames are pinned.

The number of pages reserved or used in paging space is 1128.

The number of pages allocated in the virtual space is 2394.

Information about each segment belonging to the class System is then displayed.

No distribution between the different page sizes is displayed since only 4KB pages are used by the class.

11. To print out the memory usage statistics for the superclass *default* and *developer*, type:

```
svmon -W default developer
```

12. To print out the memory usage for the superclass *Mysupclass* with its subclasses, type:

```
svmon -W Mysupclass -e
================================================================================
Superclass                          Inuse        Pin       Pgsp    Virtual
Mysupclass                           1128         14          0        596
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1128         14          0        596
================================================================================
Class                               Inuse        Pin       Pgsp    Virtual
Mysupclass.Default                   1128         14          0        596
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1128         14          0        596
  Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
  f9bf         - pers /dev/hd2:6386               s    123     0    -       -
  6fcd         - work                             s     81     2    0      79
  d1da         - work                             s     79     2    0      77
   260         - work                             s     77     2    0      75
  ea5d         - work                             s     77     2    0      75
  3fe7         - work                             s     66     2    0      64
  d7da         - work                             s     66     2    0      64
  87d0         - work                             s     66     2    0      64
  3867         - pers /dev/hd2:4205               s     57     0    -       -
  522a         - pers /dev/hd2:4392               s     47     0    -       -
  1a23         - pers /dev/hd2:24764              s     42     0    -       -
================================================================================
Class                               Inuse        Pin       Pgsp    Virtual
Mysupclass.Shared                       0          0          0          0
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB          0          0          0          0
================================================================================
Class                               Inuse        Pin       Pgsp    Virtual
Mysupclass.subclass                     0          0          0          0
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB          0          0          0          0
```

As **-e** flag is specified, the statistics of the superclass **Mysupclass** are reported with the statistics of its subclasses.

13. To print out the memory usage for all defined tiers, type:

```
svmon -T
================================================================================
Tier                                Inuse        Pin       Pgsp    Virtual
   0                                58740      28140       1083      46875
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB      50548      19948       1083      38683
```

```
    L  16 MB          2          2          0          2
================================================================================
Superclass                          Inuse        Pin       Pgsp    Virtual
System                              52565      28095       1083      43705
Shared                               3097          0          0       3097
Unclassified                         3078         45          0         73
Unmanaged                               0          0          0          0
Default                                 0          0          0          0
================================================================================
Tier                                Inuse        Pin       Pgsp    Virtual
   1                                 1128         14          0        596
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1128         14          0        596
    L  16 MB          0          0          0          0
================================================================================
Superclass                          Inuse        Pin       Pgsp    Virtual
Mysupclass                           1128         14          0        596
```

All the superclasses of all the defined tiers are reported.

14. To print out the memory usage for the tier 0, type:

```
svmon -T 0
```

```
================================================================================
Tier                                Inuse        Pin       Pgsp    Virtual
   0                                58740      28140       1083      46875
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB      50548      19948       1083      38683
    L  16 MB          2          2          0          2
================================================================================
Superclass                          Inuse        Pin       Pgsp    Virtual
System                              52565      28095       1083      43705
Shared                               3097          0          0       3097
Unclassified                         3078         45          0         73
Unmanaged                               0          0          0          0
Default                                 0          0          0          0
```

15. To print out the memory usage for the tier 0 subclasses of the superclass *Mysupclass*, type:

```
svmon -T 0 -a Mysupclass
```

```
================================================================================
Tier Superclass                     Inuse        Pin       Pgsp    Virtual
   0 Mysupclass                      1129         14          0        596
     PageSize      Inuse        Pin       Pgsp    Virtual
     s   4 KB       1129         14          0        596
================================================================================
Class                               Inuse        Pin       Pgsp    Virtual
Mysupclass.Default                   1129         14          0        596
Mysupclass.Shared                       0          0          0          0
Mysupclass.subclass                     0          0          0          0
```

All the subclasses, belonging to the tier 0, of the superclass *Mysupclass,* are reported.

16. To print out the memory usage for the tiers 0 and 1, including the segment statistics, type:

```
svmon -T 0 1 -x
```

17. To print out the memory usage for the tier 1, including the subclasses statistics, type:

```
svmon -T 1 -e
```

```
================================================================================
Tier                                Inuse        Pin       Pgsp    Virtual
   1                                 1129         14          0        596
    PageSize      Inuse        Pin       Pgsp    Virtual
    s   4 KB       1129         14          0        596
================================================================================
Superclass                          Inuse        Pin       Pgsp    Virtual
Mysupclass                           1129         14          0        596
================================================================================
Class                               Inuse        Pin       Pgsp    Virtual
```

```
Mysupclass.Default                    1129         14          0        596
Mysupclass.Shared                        0          0          0          0
Mysupclass.subclass                      0          0          0          0
```

Each superclass belonging to the tier 1 is reported with its subclass statistics.

18. To print out the memory usage for the subclasses, in the tier 0, of the superclass *Mysupclass*, including the segment statistics and the list of process identifiers, type:

```
svmon -T 0 -a Mysupclass -x -l
```

```
================================================================================
Tier Superclass                      Inuse        Pin       Pgsp    Virtual
   0 Mysupclass                       4152       4118          0       4152
     PageSize      Inuse        Pin       Pgsp    Virtual
     s   4 KB         56         22          0         56
     L  16 MB          1          1          0          1
================================================================================
Class                                Inuse        Pin       Pgsp    Virtual
Mysupclass.Default                    4152       4118          0       4152
    Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
    1140  70000000 work default shmat/mmap          L      1     1    0       1
                   pid:827620   tier:0 class:Mysupclass.Default
   a9135         - work                            s     27    22    0      27
                   Unused segment
   6910d  9001000a work shared library text         s     16     0    0      16
                   Shared library text segment
   99133  8fffffff work private load                s      5     0    0       5
                   pid:827620   tier:0 class:Mysupclass.Default
   7112e  80020014 work private load                s      4     0    0       4
                   pid:827620   tier:0 class:Mysupclass.Default
   7912f  ffffffff work application stack           s      3     0    0       3
                   pid:827620   tier:0 class:Mysupclass.Default
   89131        11 work text data BSS heap          s      1     0    0       1
                   pid:827620   tier:0 class:Mysupclass.Default
================================================================================
Class                                Inuse        Pin       Pgsp    Virtual
Mysupclass.Shared                        0          0          0          0
================================================================================
Class                                Inuse        Pin       Pgsp    Virtual
Mysupclass.subclass                      0          0          0          0
```

The statistics of all the subclasses, in the tier 0, of the superclass Mysupclass are reported.

Then, because a 16MB page pool is defined on the machine, the distribution between the different page sizes is displayed.

Then, as **-x** is specified, the subclass statistics are followed by its segments statistics.

Finally, as **-l** is specified, for each segment, the list of process which uses it, is displayed.

In this list, the process identifier followed by its tier number and its class name are given.

19. To print out the memory usage statistics for the process 278620, type:

```
 svmon -P 278620
--------------------------------------------------------------------------------
   Pid Command          Inuse        Pin     Pgsp    Virtual 64-bit Mthrd  16MB
278620 shm_lgpg64       13781      11319     1083      13769      Y     N     Y
   PageSize      Inuse        Pin       Pgsp    Virtual
   s   4 KB       5589       3127       1083       5577
   L  16 MB          2          2          0          2
  Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
 1cf8e  70000000 work default shmat/mmap          L      2     2    0       2
     0         0 work kernel segment             s   5025  3127 1083    5025
 1a9ad  90000000 work shared library text         s    420     0    0     420
 20cf0  90020014 work shared library text         s     96     0    0      96
 18f8c  9001000a work shared library text         s     16     0    0      16
 16d0b  9fff ffff pers ,/dev/hd2:309263           s     10     0    -       -
 30af8  9ffffffe work                            s      7     0    0       7
  ef87  8fffffff work private load                s      5     0    0       5
```

```
 1af8d  80020014 work private load                s      4     0    0       4
 10f88  ffffffff work application stack           s      3     0    0       3
 16f8b        10 pers text data BSS heap          s      2     0    -       -
                 /dev/hd1:2053
 14f8a        11 work text data BSS heap          s      1     0    0       1
```

The process 278620 is running the **shm_lgpg64** command. It is a 64-bit process, not a multithreaded process, and is a process using 16MB pages.

This process uses 13781 pages in real memory. 11319 frames are pinned.

The number of pages reserved or used in paging space is 1083.

The number of pages allocated in the virtual space is 13769.

Within these pages, this process uses 2 pages of 16MB (which are equivalent to 8192 pages of 4KB).

Information about each segment belonging to the process is then displayed.

The segment 1cf8e uses 16MB pages.

20. To print out the memory usage statistics for processes 6746 and 10078 taking into account only working and persistent segments, type:

    ```
    svmon -P 6746 10078 -wf
    ```

21. To print out the memory usage statistics of the top 10 processes according to the number of pinned pages, enter

    ```
    svmon -P -t 10 -p
    ```

22. To print out the memory usage statistics of all of the processes taking into account only non-system working segments and sorting the processes by the number of virtual pages, type:

    ```
    svmon -P -n -w -v
    ```

23. To print out the memory usage statistics of segments 700e 8811 and bed7, type:

    ```
    svmon -S 700e 8811 bed7 1cf8e
      Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
      700e         - work kernel pinned heap           s   1236   460 1128    1638
      8811         - work                              s    379     0   26    1757
      bed7         - pers /dev/hd2:41146               s      0     0    -       -
     1cf8e         - work                              L      2     2    0       2
    ```

    Information about each segment of the list is displayed.

    Esid is not provided because this field is valid only when the segment belongs to address space of the process.

    The segment 700e is the kernel pinned heap.

    The segment bed7 is relative to a file whose device is /dev/hd2 and whose inode number is 41146.

    The segment 8811 has no special characteristics.

    The Paging space and Virtual fields of the segment bed7 are not meaningful (persistent segment).

    The segment 1cf8e is a 16MB page segment that contains 2 pages of 16MB (equivalent to 8192 pages of 4KB).

24. To print out the memory usage statistic of segments e00e and 15015, result sorted by the number of reserved paging space blocks, type:

    ```
    svmon -S e00e 15015 -g
    ```

25. To print out the memory usage statistics of the top 5 working segment according to the number of virtual pages , type:

    ```
    svmon -S -t 5 -w -v
    ```

26. To sort system segments by the number of pages in real memory, and print out the top 10 system segments of the resulting list, type:

    ```
    svmon -S -s -u -t 10
    ```

27. To print only 16MB page segments, type:

    ```
    svmon -S -q L
    ```

28. To display the address ranges of the segment 30138, type:

```
svmon -S 30138 -r
     Vsid      Esid Type Description              PSize  Inuse   Pin    Pgsp Virtual
     30138         - work page frame table            s  10752   10752  0    10752
                     Addr Range: 0..10751 : 0..65535
```

29. To display the file path the persistent segment 60218 refers to, type:

```
svmon -S 60218 -j
    Vsid      Esid Type Description                     PSize   Inuse   Pin Pgsp Virtual
   60218         - pers large file /dev/hd9var:214   -      s     328     0    -       -
                        /var/tmp/hostmibd.log
```

30. To print out the frames belonging to the segment 36cfb type:

```
svmon -D 36cfb
Segid: 36cfb
Type:  persistent
PSize: s (4 KB)
Address Range: 0..4

           Page      Frame    Pin  ExtSegid    ExtPage
              0      62861     N          -          -
              1      70594     N          -          -
              2      24204     N          -          -
              3      46339     N          -          -
              4      23868     N          -          -
         122070      78191     N     208831       dcd6
```

The segment 36cfb is a persistent segment with 5 pages. None of them is pinned.

The page 122070 is physically the page dcd6 in the extended segment 208831.

31. To print out the frames belonging to the segment 36cfb with the status bit of each frame, type:

```
svmon -D 36cfb -b
Segid: 36cfb
Type:  persistent
PSize: s (4 KB)
Address Range: 0..5

           Page      Frame    Pin        Ref        Mod  ExtSegid    ExtPage
              0     392948     N          Y          N         -          -
              1     393187     N          Y          N         -          -
              2     392968     N          Y          N         -          -
              3     392825     N          Y          N         -          -
              4     392890     N          N          N         -          -
              5      23651     N          Y          N         -          -
```

As the **-b** flag is specified, the referenced and modified bits of each frame of the segment 36cfb are displayed.

32. To print out the frames belonging to the segment 36cfb including the maximum memory size dynamically allocated by svmon during its execution, type:

```
svmon -D 36cfb -z
Segid: 36cfb
Type:  persistent
PSize: s (4 KB)
Address Range: 0..5

           Page      Frame    Pin  ExtSegid    ExtPage
              0     392948     N          -          -
              1     393187     N          -          -
              2     392968     N          -          -
              3     392825     N          -          -
              4     392890     N          -          -
              5      23651     N          -          -
Maximum memory allocated = 916
```

33. To print out the percentage of memory accessed during a 10-second interval, type:

```
svmon -F -i 10 2
```

```
Processing.. 100%
percentage of memory used:   73.85%
percentage of 16 MB page memory used:   50.00%
Processing.. 100%
percentage of memory used:   22.83%
percentage of 16 MB page memory used:   50.00%
```

The percentage of analyzed memory is provided in order to keep the user aware of the processing.

The percentage of memory accessed during the specified interval (that means frames with the reference flag set) is 22.83%.

The percentage of 16MB page memory used (that means pages used by a segment) is 50.00%.

34. To print out the status of the frames 94072 and 672914, type:

```
svmon -F 94072 672914
      Frame  Segid Ref Mod    Pincount   State   Swbits ExtSegid PSize
      94072   e6fd   N   N     0/0      In-Use 88000004        -     s
     672914  784de   Y   Y     0/0      In-Use 88100004   250054     s
```

The frame 94072 belongs to the segment e6fd.

The frame is not referenced and not modified, it is in the In-Use state, it does not pertain to an extended segment nor to a large page segment.

The frame 672914 belongs to the extended segment 250054 that is linked to the 784de primary segment.

Related Information

The **ncheck** command, **rmss** command.

Logical volume storage and Paging space in *Operating system and device management*.

Workload management in *Operating system and device management*.

---

swap Command

Purpose

Provides a paging space administrative interface.

Syntax

swap [**-a** *device*] | [**-d** *device*] | [**-s**] | [**-l**]

Description

The functions provided by the swap command are diplay of characteristics, addition of paging space and removal of paging space.

Flags

-a *device*	Activates the paging space. Performs the same function the **swapon** command.
-d *device*	Deactivates the paging space. Performs the same function as the **swapoff** command.

-l Lists the status of paging space areas in a list form. The output has 4 columns, containing the following information:

device Path name of the page space.

maj/min
The major/minor device number for the device.

total Total size in megabytes for the area.

free Amount of available space.

-s Prints summary information about total paging space usage and availability. The following information is displayed in the output (amounts of paging space are listed in 4K byte blocks).

allocated
Total amount of paging space area currently allocated.

used Total amount of paging space area currently being used.

available
Total amount of free paging space.

These numbers include paging spaces from all configured areas as listed by the **-l** option on active paging space.
Note: There is a paging space limit of 64 GB per device.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To print summary information on total paging space, enter:
   ```
   swap -s
   ```
2. To list the status of the paging space areas in a list form, enter:
   ```
   swap -l
   ```
3. To activate a particular paging space device paging01, enter:
   ```
   swap -a /dev/paging01
   ```

Files

/usr/sbin/swap Contains the System V **swap** command.

Related Information

The **chps** command, **lsps** command, **swapoff** command, **swapon** command.

---

swapoff Command

Purpose

Deactivates one or more paging spaces.

Syntax

swapoff *DeviceName* { *DeviceName* ...}

Description

The **swapoff** command deactivates one or more paging spaces. The paging spaces are specified by *DeviceName*.

Note: There is a paging space limit of 64 GB per device.

To be deactivated:

- The paging space must have been previously activated through the **swapon** command.
- There must exist enough space in the remaining paging spaces. The remaining paging device should have enough space to accommodate the current system-wide paging space usage and the **npswarn** value.

Exit Status

Value	Description
0	Deactivation is successful, the paging state is set to the INACTIVE state.
1	The following message displays: `swapoff: Cannot deactivate paging space` *DeviceName*

2	There is not enough space in the remaining paging spaces, the deactivation is not done and the following message displays:
	`"swapoff: Cannot deactivate paging space` *DeviceName* `:`
	`There is not enough space in the file system."`

3	An I/O error occurred on user pages of a paging space, the following message displays:
	`swapoff: Deactivation of paging space` *DeviceName* `suspended:`
	`I/O errors encountered on user backing pages.`
	The recommended action is:
	• Check the error log.
	• Deactivate the paging space for the next reboot using the **chps** command.
	• Reboot the system.

4	An I/O error occurred on system pages of a paging space, the following message displays:
	`swapoff: Deactivation of paging space` *DeviceName* `suspended:`
	`I/O errors encountered on system backing pages. The system may crash.`
	The recommended action is:
	• Check the error log.
	• Deactivate the paging space for the next reboot using the **chps** command.
	• Reboot the system.

Related Information

The **lsps**, **swap**, **swapon**, and **vmo** commands.

The Paging space in *Operating system and device management*.

---

swapon Command

Purpose

Activates a paging space.

Syntax

swapon -a | *DeviceName* ...

Description

The **swapon** command activates a paging space. It is used during early system initialization to make the initial paging space available. During a later phase of system initialization, the **swapon -a** command is used to make other devices available so that paging and swapping activity is interleaved across several devices. The **swapon -a** command makes all devices specified in the **/etc/swapspaces** file available. Calls to the **swapon** command normally occur in the system multiuser initialization **/etc/rc** file.

The *DeviceName* parameter specifies a specific device to be made available. The second form gives individual block devices as given in the system swap configuration table. The call makes this space and other defined spaces available to the system for paging and swap allocation. The system swap configuration table is the set of all devices specified in the **/etc/swapspaces** file.

You can use the Devices application in Web-based System Manager (wsm) to run this command.

Note: The maximum number of active paging spaces is 16. In addition, there is a paging space limit of 64 GB per device.

Flags

-a Causes all devices present in the **/etc/swapspaces** file to be made available.

Examples

1. To cause all devices present in the **/etc/swapspaces** file to be made available, enter:

   ```
   swapon  -a
   ```

   All devices present in the **/etc/swapspaces** file are now available.
2. To cause the **/dev/paging03** and **/dev/paging04** devices to be available for paging and swapping, enter:

   ```
   swapon /dev/paging03 /dev/paging04
   ```

   The **/dev/paging03** and **/dev/paging04** devices are now available.

Files

/etc/rc	System multiuser initialization
/dev/paging	Device entries for paging/swap space
/etc/swapspaces	Contains a list of swap devices.

Related Information

The **mkps** command, **chps** command.

The **swapon** subroutine.

The Paging space in *Operating system and device management* explains paging space and its allocation policies.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

The System management interface tool in *Operating system and device management* explains the structure, main menus, and tasks that are done with SMIT.

swcons Command

Purpose

Redirects, temporarily, the system console output to a specified device or file.

Syntax

swcons [**-p** *Log_File*] [**-s** *Log_Size*] [**-t** *Tag_Verbosity*] [**-v** *Log_Verbosity*] *PathName*

Description

The **swcons** command temporarily switches the system console output to a different target during system operation. This command only switches system informational-, error-, and intervention-required message output to the specified destination. The **swcons** command does not affect the operation of the system console device that is providing a login by way of the **getty** command.

The device or file specified when using this command remains the target for console output until changed by another **swcons** command, until the next start of the system, or until the console driver detects an error when accessing the designated device or file. If an open or write error is detected on the device or file specified by the **swcons** command, the console device driver switches all output back to the device or file that provided console support when the system was last started.

The *PathName* parameter must be a fully qualified path name to a device or file that is to receive system console message output. If the *PathName* parameter specifies a file that does not exist, the **swcons** command creates the file. If the file does exist, the **swcons** command appends any new console message output to the contents of the file.

> **Attention:** Use of the **swcons** command to switch console output to an NFS mounted file system may cause the operating system to hang.

Flags

-p *Log_File*	Specifies the full path name to use for the console output log file.
-s *Log_Size*	Specifies the size, in bytes, of the console output log file.
-t *Tag_Verbosity*	Specifies the verbosity level for console output tagging. Zero disables tagging; 1 through 9 enable tagging. For additional information about console output logging and tagging, see the console Special File in the *AIX 5L Version 5.3 Files Reference* book.
-v *Log_Verbosity*	Specifies the verbosity level for console output logging. Zero disables logging; 1 through 9 enable logging.

Examples

1. To change the system console message output to a file called `console.out` in the `/tmp` directory, enter:
   ```
   swcons /tmp/console.out
   ```
2. To change the system console message output to a terminal with the logical name `tty3`, enter:
   ```
   swcons /dev/tty3
   ```
3. To change the system-console message output back to the device or file that supported the console output at system start time, enter:
   ```
   swcons
   ```

Files

/dev/console	Specifies the special file for system console access.
/usr/sbin/swcons	Contains the **swcons** command file.

Related Information

The **chcons** command, **lscons** command.

The **console** special file.

---

swts Command

Purpose

Switches a thin server to a different COSI.

Syntax

swts -c *Image* [**-t** *cron_time*] [**-v**] *ThinServer*

Description

The **swts** command switches a thin server to a different Common Operating System Image (COSI). If specified with the **-t** flag, the thin server switches to a new common image at the time specified by the *Time* parameter. The value for *Time* must be a valid cron tab entry. Refer to the **crontab** command for creating valid cron time entries.

The **swts** command can be run on either a NIM master or a thin server. When a thin server is switched to a new common image, files in the **/inst_root** directory for the thin server will be synced with the new common image.

Flags

-c *Image*	Specifies the common image that the thin server switches to.
-t *cron_time*	Specifies a cron entry that allows thin servers to be switched over at a more convenient time.
-v	Enables verbose debug output when the **swts** command runs.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **swts** command.

Examples

1. To switch the `cosi1` common image of a thin server named `lobo` to a common image named `cosi2`, enter:

   ```
   swts -c cosi2 lobo
   ```

The `lobo` thin server is re-initialized and `cosi2` is its new operating system.

2. To switch the `cosi1` common image of a thin server named `lobo` to a common image named `cosi2` at midnight on Sunday, December 25, enter:

   ```
   swts -c cosi2 -t "0 0 25 12 0" lobo
   ```

   The `lobo` thin server will continue to use the `cosi1` common image until midnight on Sunday, December 25, when it switches to `cosi2`.

Location

/usr/sbin/swts

Files

/etc/niminfo	Contains variables used by NIM.

Related Information

The **crontab** command, **dbts** command, **lsts** command, **mkcosi** command, **mkts** command, **nim** command, **nim_clients_setup** command, **nim_master_setup** command, **nimconfig** command, **rmts** command.

---

sync Command

Purpose

Updates the i-node table and writes buffered files to the hard disk.

Syntax

sync

Description

The **sync** command runs the **sync** subroutine. If the system must be stopped, run the **sync** command to ensure file system integrity. The **sync** command writes all unwritten system buffers to disk including modified i-nodes, delayed block I/O, and read-write mapped files.

> **Note:** The writing, although scheduled, is not necessarily complete upon return from the **sync** subroutine.

Related Information

The **sync** subroutine.

---

synclvodm Command

Purpose

Synchronizes or rebuilds the logical volume control block, the device configuration database, and the volume group descriptor areas on the physical volumes.

Syntax

synclvodm -v -P *VolumeGroup LogicalVolume ...*

Description

During normal operations, the device configuration database remains consistent with the logical volume manager information in the logical volume control blocks and the volume group descriptor areas on the physical volumes. If for some reason the device configuration database is not consistent with Logical Volume Manager information, the **synclvodm** command can be used to resynchronize the database. The volume group must be active for the resynchronization to occur (see **varyonvg**). If logical volume names are specified, only the information related to those logical volumes is updated.

> **Attention:** Do not remove the **/dev** entries for volume groups or logical volumes. Do not change the device configuration database entries for volume groups or logical volumes using the object data manager.
>
> **Note:** To use this command, you must either have root user authority or be a member of the **system** group.

Flags

-P Preserves the ownership and permissions of the logical volume special files. If this flag is not set the logical volume special file ownership will be set to `root` and the group will be set to `system`.

-v Specifies that the output from the **synclvodm** command is displayed in verbose mode.

Examples

To synchronize the device configuration database with the logical volume manager information for rootvg, enter the following:

```
synclvodm rootvg
```

Files

/usr/sbin/synclvodm	Contains the **synclvodm** command.

Related Information

The **varyonvg** command, **varyoffvg** command.

---

syncvg Command

Purpose

Synchronizes logical volume copies that are not current.

Syntax

syncvg [**-f**] [**-i**] [**-H**] [**-P** *NumParallelLps*] { **-l** | **-p** | **-v** } *Name ...*

Description

The **syncvg** command synchronizes the physical partitions, which are copies of the original physical partition, that are not current. The **syncvg** command can be used with logical volumes, physical volumes, or volume groups, with the *Name* parameter representing the logical volume name, physical volume name, or volume group name. The synchronization process can be time consuming, depending on the hardware characteristics and the amount of data.

When the **-f** flag is used, a good physical copy is chosen and propagated to all other copies of the logical partition, whether or not they are stale. Using this flag is necessary in cases where the logical volume does not have the mirror write consistency recovery.

Unless disabled, the copies within a volume group are synchronized automatically when the volume group is activated by the **varyonvg** command.

Note: For the **sycnvg** command to be successful, at least one good copy of the logical volume should be accessible, and the physical volumes that contains this copy should be in ACTIVE state. If the **-f** option is used, the above condition applies to all mirror copies.

If the **-P** option is not specified, **syncvg** will check for the *NUM_PARALLEL_LPS* environment variable. The value of *NUM_PARALLEL_LPS* will be used to set the number of logical partitions to be synchronized in parallel.

Flags

-f	Specifies a good physical copy is chosen and propagated to all other copies of the logical partition, whether or not they are stale.
-H	Postpones writes for this volume group on other active concurrent cluster nodes until this sync operation is complete. When using the **-H** flag, the **-P** flag does not require that all the nodes on the cluster support the **-P** flag. This flag is ignored if the volume group is not varied on in concurrent mode.
-i	Reads the names from standard input.
-l	Specifies that the *Name* parameter represents a logical volume device name.
-p	Specifies that the *Name* parameter represents a physical volume device name.
-P *NumParallelLps*	Numbers of logical partitions to be synchronized in parallel. The valid range for *NumParallelLps* is 1 to 32. *NumParallelLps* must be tailored to the machine, disks in the volume group, system resources, and volume group mode. When a volume group is varied on in concurrent mode, all other cluster nodes that have this volume group varied must be at least AIX 4.3.0, otherwise **syncvg** will ignore this option and continue. **Note:** See Description above for more information.
-v	Specifies that the *Name* parameter represents a volume group device name.

Examples

1. To synchronize the copies on physical volumes `hdisk04` and `hdisk05`, enter:

   ```
   syncvg  -p hdisk04 hdisk05
   ```

2. To synchronize the copies on volume groups `vg04` and `vg05`, enter:

   ```
   syncvg  -v vg04 vg05
   ```

Files

/usr/sbin/syncvg	Contains the **syncvg** command.
/tmp	Directory where the temporary files are stored and while the command is running.

Related Information

The **varyonvg** command.

The Logical volumn storage in *Operating system and device management* explains the Logical Volume Manager, physical volumes, logical volumes, volume groups, organization, ensuring data integrity, and allocation characteristics.

The System management interface tool in *Operating system and device management* explains the structure, main menus, and tasks that are done with SMIT.

syscall Command

Purpose

Performs a specified subroutine call.

Syntax

syscall [**-n**] *Name* [*Argument1 ... ArgumentN*] [**;** *Name* [*Argument1 ... ArgumentN*]] ...

Description

The **syscall** command executes a system call interface program, which performs the subroutine call specified by the *Name* parameter. If you specify the **-n** flag, the **syscall** command performs the call **n** times. Arguments specified by the *Argument* parameter are passed to the subroutine without error checking. The *Argument* parameter can be expressed in the following formats:

0x *nnn*	Hexadecimal constant *nnn*.
0 *nnn*	Octal constant *nnn*.
nnn	Decimal constant *nnn*.
+*nnn*	Decimal constant *nnn*.
-*nnn*	Decimal constant *nnn*.
"string	The character string *"string"*.
'*string*	The character string *"string"*.
string	The character string *"string"*.
#*string*	The length of the character string *"string"*.
&&*n*	The address of the *n*th argument to this subroutine. (*n*=0 is the subroutine name.)
&*n*	The address of the *n*th byte in an internal 10KB buffer.
$*n*	The result of the *n*th subroutine. (*n*=0 is the first subroutine.)
string	Anything else is a literal character string.

The **syscall** command prints a message and exits for unknown subroutines and for subroutines that return a value of -1.

Note: The **syscall** command understands the **sleep** subroutine as a special case subroutine.

Flags

-n Specifies the number of times the **syscall** command performs the specified subroutine.

; Separates multiple subroutines (up to a maximum of 20) issued by the same invocation of the **syscall** command.

Examples

To simulate the C program fragment:

```
output=open("x", 401, 0755);
write(output, "hello", strlen("hello"));
```

enter:

```
syscall open x 401 0755 \; write \$0 hello \#hello
```

Note: Special shell characters must be escaped.

Files

/usr/bin/syscall Contains the **syscall** command.

Related Information

The **bsh** command, **Rsh** command, **csh** command, **ksh** command, **sh** command.

The **open** subroutine, **sleep** subroutine.

Shells in *Operating system and device management*.

---

sysck Command

Purpose

Checks the inventory information during installation and update procedures.

Syntax

sysck { **-i** | **-u** } [**-R** *RootPath*] [**-N**] [**-v**] [**-s** *SaveFile*] [**-O** { **r** | **s** | **u** }] **-f** *File ProductName* { **tcbck** *Flags* }

All of the **tcbck** command flags are valid with this command.

Description

> **Attention:** The **sysck** command DOES NOT support checking files that are greater than 2 gigabytes. If a product needs to ship a file that is greater than 2 gigabytes, set size and checksum values in their *Fileset*.inventory to **VOLATILE**, so the **sysck** command will not try to access the file.
>
> **Note:** All of the **tcbck** command flags are valid with the **sysck** command. This feature provides compatibility with Version 3.1. For more information on the **tcbck** command and a complete listing of its flags, refer to *AIX 5L Version 5.3 Commands Reference*.

The **sysck** command checks file definitions against the extracted files from the installation and update media and updates the Software Vital Product Data (SWVPD) database. The **sysck** command does not recognize the following special characters in file names: `, ', \, ″, ^, ( ), |, {}, [], <>,and : . If a file name contains one of these characters, the **sysck** command fails.

The **sysck** command is primarily used during the installation and update of software products.

When invoked with the **-i** flag, the **sysck** command checks the attributes of an extracted file with its file definitions, updates the SWVPD, and attempts to fix some errors if they exist.

The *File* parameter is the name of the stanza file that contains the file definitions. An example of such a file is the **/etc/security/sysck.cfg** file, although the **syschk** command does not use this file. The **sysck** command checks the size, links, symlinks, owner, group, and mode attributes of a file for which the type attribute is set to **FILE**. When invoked with the **-v** flag as well as the **-i** flag, **sysck** also checks the checksum value of a file.

The **sysck** command updates the file name, product name, type, checksum, and size of each file in the SWVPD database.

To fix errors, the **sysck** command resets the attribute of the installed or updated file to the defined value in the *File* stanza file, except for some attributes as described in ″Fixing Errors″.

When invoked with the **-u** flag, the **sysck** command removes the entry from the SWVPD database for each file that is part of the software product *ProductName*. The **sysck** command also deletes any hard links and symbolic links for each file, as defined in the SWVPD database.

Flags

-f *File* Specifies the name of the stanza file that contains the file definitions.

-i Checks for the correct installation of a software product's files. Updates the SWVPD database with the file definitions, and attempts to fix some errors if found.

-N Specifies that the SWVPD database should not be updated.

-O {**r**|**s**|**u**} Specifies which part of the SWVPD is to be updated, as follows:

- **r** Specifies the root part of the SWVPD.
- **s** Specifies the **/usr/share** part of the SWVPD.
- **u** Specifies the **/usr** part of the SWVPD (default).

-R *RootPath* Use *RootPath* as root instead of *"/"*.

-s *SaveFile* Takes a snapshot of what is currently in the VPD and saves it in stanza format to the file specified by *SaveFile*. Called with the **-u** option. No action is taken in the database with this flag. Must be used with the **-f** option. For example:

```
sysck -i -s /tmp/save.inv -f /tmp/real.inv bos.rte.shell
```

-u Deletes file entries from the SWVPD and deletes hard links and symbolic links.

-v Verifies that the checksum is correct.

ProductName Specifies the installable software product or option that is being checked.

Environment Variables

INUTREE The environment variable **INUTREE** has only the following four valid values:

- **NULL** Same as **INUTREE** not being set.
- **M** Specifies the root part of the SWVPD.
- **S** Specifies the **/usr/share** part of the SWVPD.
- **U** Specifies the **/usr** part of the SWVPD (default).

INUTREE can be used instead of the **-O** *Tree* flag.

INUNOVPD The environment variable **INUNOVPD** can be null or can be set to 1. If it is set to 1 then **sysck** does not update the SWVPD. **INUNOVPD** can be used instead of the **-N** flag.

INUVERIFY If the environment variable **INUVERIFY** is set to 1 **sysck** verifies that the checksum attributes in the stanza file are correct. **INUVERIFY** can be used instead of the **-v** flag.

File Definitions

acl The access control list for the file. If the value is blank, the **acl** attribute is removed. If no value is specified, the command computes a value, according to the format described in Access Control Lists.

This attribute should grant x (execute) access only to the root user and members of the security group. The command should **setuid** to the root user and have the trusted computing base attribute.

class The logical group of the file. A value must be specified because it cannot be computed. The value is *ClassName* [*ClassName*].

checksum The checksum of the file. If the value is blank, the **checksum** attribute is removed. If no value is specified, the command computes a value, according to the format given in the **sum** command. The value is the output of the **sum -r** command, including spaces.

group The file group. If the value is blank, the **group** attribute is removed. If no value is specified, the command computes a value, which can be a group ID or a group name.

mode The file mode. If the value is blank, the **mode** attribute is removed. If no value is specified, the command computes a value, which can be an octal number or a string (**rwx**), and have the **TCB**, **SUID**, **SGID**, and **SVTX** attributes.

owner The file owner. If the value is blank, the **owner** attribute is removed. If no value is specified, the command computes a value, which can be a user ID or a user name.

size The size of the file in bytes. If the value is blank, the **size** attribute is removed. A **VOLATILE** value in the size field indicates that the file size will change (so no checksum value can be given). A **NOSIZE** value indicates that the file has 0 length. If no value is specified, the command computes a value, which is a decimal number.

target Allows symbolic links and hard links to exist as separate stanzas in the inventory. The **target** file definition refers to the full path name of the source of the link, for example:

```
/etc/foo --> /usr/bar
```

The **target** is `/usr/bar`.

type The type of file. This value cannot be blank. If no value is specified, the command computes a value, which can be the **FILE**, **DIRECTORY**, **FIFO**, **BLK_DEV**, **CHAR_DEV**, **LINK**, **MPX_DEV**, and **SYMLINK** keywords.

xacl An addition to the extended-access control list. A value must be specified as a single entry in an extended-access control list because the value cannot be computed. This attribute is valid only if the **-i** flag is used. For information about the format, see the **acl** file definition above.

Fixing Errors

To fix errors, the **sysck** command resets the attribute of the installed or updated file to the defined value defined in the *File* stanza file except for the following attributes, for which the **sysck** command acts as described:

links Creates any missing hard links. If a link exists to another file that is not listed in this definition, the link is deleted.

program If this attribute is included in the *File* stanza file, **sysck** invokes the program. A message is printed if an error occurs, but no additional action is taken.

symlinks Creates any missing symbolic links. If a link exists to another file that is not listed in this definition, the link is deleted.

Security

Privilege Control: Only the root user can run this command.

Examples

1. A product that uses the **installp** command to install ships an inventory file in its image. To add the definitions to the inventory database and check permissions, links, checksums, etc., enter:

   ```
   sysck  -i  -f dude.rte.inventory dude.rte
   ```

   where `dude.rte.inventory` would look like the following:

   ```
   /usr/bin/dude.exec:
      class = apply,inventory,dude.rte
      owner = bin
      group = bin
      mode = 555
      type = FILE
      size = 2744
      checksum = "04720        3"
   ```

2. To remove any links to files for a product that has been removed from the system and remove the files from the inventory database, enter:

   ```
   sysck  -u  -f dude.rte.inventory dude.rte
   ```

Files

/etc/objrepos/inventory	Specifies names and locations of files in a software product on the root.
/usr/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr** file system.
/usr/share/lib/objrepos/inventory	Specifies names and locations of files in a software product on the **/usr/share** file system.

Related Information

The **installp** command, **sum** command, **tcbck** command.

---

syscorepath Command

Purpose

Specifies a single system-wide directory where all core files of any processes will be dumped.

Syntax

syscorepath [**-p** *DirectoryName*] [**-g**] [**-c**]

Description

The **syscorepath** command enables a system administrator to set up a single system-wide directory in which to dump core files from any processes. This can ease administrative tasks in managing file-system space and provides a single, known directory in which to find core files. By default, the core file is created in the working directory of the process being core-dumped.

The directory should have read and write privileges for all users on the system. If a user does not have permission to write in the directory, a core file will not be created. Core files will be given unique names based on the process ID and time, so a core file will be named **core.***pid.ddhhmmss*, where *pid* is the process ID, *dd* is the day of the month, *hh* is the hour in 24-hour format, *mm* is minutes, and *ss* is seconds.

Flags

-c	Unsets the current directory specified as the repository for core files. Subsequent core files will be created in the working directory of the process.
-g	Displays current directory specified as the repository for core files.
-p *DirectoryName*	Specifies the directory to use as a repository for core files. *DirectoryName* must be a valid directory name.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Standard Errors

EPERM

User does not have permission.

ENOTDIR
Specified *DirectoryName* is not a directory.

ENAMETOOLONG
Specified *DirectoryName* is too long.

Security
Only the root user can run this command.

Examples
1. To set **/core** as the repository for core files, type:
   ```
   syscorepath -p /core
   ```
2. To display the current repository for core files, type:
   ```
   syscorepath -g
   ```
3. To unset the directory used as the repository for core files, type:
   ```
   syscorepath -c
   ```

Files

/usr/bin/syscorepath	Contains the **syscorepath** command.

Related Information
The **dbx** command, **gencore** command.

The **a.out** file format, **core** file format.

---

sysdumpdev Command

Purpose
Changes the primary or secondary dump device designation in a running system.

Syntax
sysdumpdev -P { **-p** *Device* | **-s** *Device* } [**-q**] [**-i**]

sysdumpdev [**-p** *Device* | **-s** *Device*] [**-q**]

sysdumpdev [**-d** *Directory* | **-D** *Directory* | **-e** | **-l** | [**-k** | **-K**] | **-l** | **-p** *Device* | **-q** | **-r** *Host***:** *Path* | **-s** *Device* | **-z**]

sysdumpdev [**-c** | **-C**] [**-i**]

sysdumpdev -L { **-v** | **-S** *Device* }

Description
The **sysdumpdev** command changes the primary or secondary dump device designation in a system that is running. The primary and secondary dump devices are designated in a system configuration object. The new device designations are in effect until the **sysdumpdev** command is run again, or the system is restarted.

If no flags are used with the **sysdumpdev** command, the dump devices defined in the **SWservAt** ODM object class are used. The default primary dump device is **/dev/hd6**. The default secondary dump device is **/dev/sysdumpnull**.

Notes:

1. A mirrored paging space may be used as a dump device.
2. Do not use a diskette drive as your dump device.
3. If you use a paging device, only use hd6, the primary paging device. AIX 4.2.1 or later supports using any paging device in the root volume group (rootvg) as the secondary dump device.
4. If you use a removable device such as a tape or DVD, be aware that the dump does not span volumes. thus, the dump must fit on a single volume.

You can use the Devices application in Web-based System Manager (wsm) to run this command.

You can also use the **sysdumpdev** command to specify whether or not dumps should be compressed before writing them to the dump device. Compressing dumps reduces the size needed for dump devices, but may cause the dump process to take longer.

Notes:

1. The **savecore** command should be used to copy a compressed dump from the dump device to a file.
2. The dump compression feature only applies to AIX 4.3.2 and later versions.
3. Dump compression is turned on by default in AIX 5.3.

Running sysdumpdev in Non-rootvg Volume Groups

You can use a dump logical volume outside the root volume group, if it is not a permanent dump device. For example, if the **-P** flag is not specified. However, if you choose a paging space, we cannot copy the dump device unless it is in rootvg. During the time we must copy the dump device, only rootvg is active before paging is started.

The primary dump devices must always be in the root volume group for permanent dump devices. The secondary device may be outside the root volume group unless it is a paging space.

Configuring Remote Dump Devices with sysdumpdev

The **sysdumpdev** command can also be used to configure remote dump devices. The following conditions must be met before a remote dump device can be configured:

- The local host must be a uni-processor machine.
- The local and the remote host must have Transmission Control Protocol/Internet Protocol (TCP/IP) installed and configured.
- The local host must have Network File System (NFS) installed.
- The remote host must support NFS.
- The remote host must be operational and on the network. This condition can be tested by issuing the **ping** command.
- The remote host must have an NFS exported directory defined such that the local host has read and write permissions as well as root access to the dump file on the remote host.
- The remote host cannot be the same as the local host.

The network device driver must support remote dump. Drivers which support remote dump include the drivers for these network devices:

- Integrated Ethernet MCA Adapter through AIX 5.1 only
- IBM 10/100 Mbps Ethernet TX MCA Adapter (8f62) through AIX 5.1 only
- IBM PCI Ethernet Adapter (22100020)

- IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
- FDDI MCA Adapter through AIX 5.1 only
- ISA Token Ring Adapter
- Token-Ring High-Performance Adapter (8fa2)
- Token-Ring High-Performance Adapter (8fc8)
- IBM PCI Token-Ring Adapter (14101800)
- IBM PCI Token-Ring Adapter (14103e00)

Drivers that do not support remote dump include the drivers for these network devices:

- ISA Ethernet Adapter
- Ethernet High-Performance LAN Adapter (8ef5)
- Ethernet High-Performance LAN Adapter (8f95)
- Gigabit Ethernet-SX PCI Adapter (14100401)

Flags

-c Specifies that dumps will not be compressed. The **-c** flag applies to only AIX 4.3.2 and later versions.

-C Specifies that all future dumps will be compressed before they are written to the dump device. The **-C** flag applies to only AIX 4.3.2 and later versions.

-d *Directory* Specifies the *Directory* the dump is copied to at system boot. If the copy fails at boot time, the **-d** flag ignores the system dump.

-D *Directory* Specifies the *Directory* the dump is copied to at system boot. If the copy fails at boot time, using the **-D** flag allows you to copy the dump to an external media.

Note: When using the **-d** *Directory* or **-D** *Directory* flags, the following error conditions are detected:

- *Directory* does not exist.
- *Directory* is not in the local journaled file system.
- *Directory* is not in the **rootvg** volume group.

-e Estimates the size of the dump (in bytes) for the current running system. If the dump will be compressed, then the size shown is the estimate of the size after compression.

-i Indicates that the **sysdumpdev** command was called from a system function. This flag is only used by system utilities. The **-i** flag will not make the requested change if the effected value has already been modified by other than an automatic IBM function; that is, the **-i** flag will not override a previous change.

-l Resets the indications of previous changes. After the **-l** flag is specified, changes are allowed with the **-i** flag.

-k If your machine has a key mode switch, it is required to be in the service position before a dump can be forced with the dump key sequences. Prior to AIX 5.3, this restriction also applies to the reset button.

-K If your machine has a key mode switch, the reset button or the dump key sequences will force a dump with the key in the normal position, or on a machine without a key mode switch.
Note: On a machine without a key mode switch, a dump can not be forced with the key sequence without this value set. This is also true of the reset button prior to AIX 5.3.

-l Lists the current value of the primary and secondary dump devices, copy directory, and **forcecopy** attribute.

-L	Displays statistical information about the most recent system dump. This includes date and time of last dump, number of bytes written, and completion status. If the dump was compressed, then this flag shows both the original uncompressed size and the compressed size of the dump. The compressed size is the size of what was actually written to the dump device. If no previous dump was recorded in nonvolatile memory, this flag scans the dump devices for the existing dump. **Note:** The dump sizes shown might not reflect the exact size of the dump on the media. There can be a small difference because of disk and copy block sizes.
-P	Makes permanent the dump device specified by **-p** or **-s** flags. The **-P** flag can only be used with the **-p** or **-s** flags.
-p *Device*	Device temporarily changes the primary dump device to the specified device. The device can be a logical volume, writable DVD, or a tape device. For a network dump, the device can be a host name and a path name.
-q	Suppresses all messages to standard output. If this flag is used with the **-l**, **-r**, **-z** or **-L** flag, the **-q** command will be ignored.
-r *Host*:*Path*	Frees space used by the remote dump file on server *Host*. The location of the dump file is specified by the *Path*.
-s *Device*	Device Temporarily changes the secondary dump device to the specified device. The same devices valid for the **-p** flag are valid here.
-S *Device*	Scans a specific dump device for a valid compressed dump. The dump must be from an AIX release with parallel dump support. This flag can be used only with the **-L** flag.
-v	When the dump status is not 0, this option will display available dump debug information. The debug data, when available, is used by service to diagnose dump failures. This flag can only be used with the **-L** flag.
-z	Determines if a new system dump is present. If one is present, a string containing the size of the dump in bytes and the name of the dump device will be written to standard output. If a new system dump does not exist, nothing is returned. After the **sysdumpdev -z** command is run on an existing system dump, the dump will no longer be considered recent.

If no flags are used with the **sysdumpdev** command, the default dump devices are used.

Security

Access Control: Only the root user can run this command.

Examples

1. To display current dump device settings, enter:

   ```
   sysdumpdev -l
   ```

2. To designate logical volume `hd7` as the primary dump device, enter:

   ```
   sysdumpdev -p /dev/hd7
   ```

3. To designate tape device `rmt0` as the secondary dump device, enter:

   ```
   sysdumpdev -s /dev/rmt0
   ```

4. To display information from the previous dump invocation, enter:

   ```
   sysdumpdev -L
   ```

   Note: A nonzero dump status indicates a failed dump. The possible dump status values and their corresponding LED values are as follows:

Dump status	Description	LED value
0	Dump completed successfully	`0C0`
-1	No dump device defined	`0C8`
-2	Dump device too small	`0C4`

Dump status	Description	LED value
-3	Dump crashed or did not start	0C5
-4	I/O error	0C1

5. To permanently change the database object for the primary dump device to /dev/newdisk1, enter:

   ```
   sysdumpdev  -P  -p /dev/newdisk1
   ```

6. To determine if a new system dump exists, enter:

   ```
   sysdumpdev  -z
   ```

   If a system dump has occurred recently, output similar to the following will appear:

   ```
   4537344 /dev/hd7
   ```

7. To designate remote dump file /var/adm/ras/systemdump on host mercury for a primary dump device, enter:

   ```
   sysdumpdev  -p mercury:/var/adm/ras/systemdump
   ```

   A : (colon) must be inserted between the host name and the file name.

8. To specify the directory that a dump is copied to after a system crash, if the dump device is /dev/hd6, enter:

   ```
   sysdumpdev  -d /tmp/dump
   ```

   This attempts to copy the dump from /dev/hd6 to /tmp/dump after a system crash. If there is an error during the copy, the system continues to boot and the dump is lost.

9. To specify the directory that a dump is copied to after a system crash, if the dump device is /dev/hd6, enter:

   ```
   sysdumpdev  -D /tmp/dump
   ```

   This attempts to copy the dump from /dev/hd6 to the /tmp/dump directory after a crash. If the copy fails, you are prompted with a menu that allows you to copy the dump manually to some external media.

10. To scan a dump device for a dump, enter:

    ```
    sysdumpdev -L -S /dev/hd6
    ```

Related Information

The **mount** command, **ping** command, **savecore** command, **sysdumpstart** command.

System Dump Facility in the *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

sysdumpstart Command

Purpose

Provides a command line interface to start a kernel dump to the primary or secondary dump device.

Syntax

sysdumpstart { **-p** | **-s** [**-f**] }

Description

The **sysdumpstart** command provides a command line interface to start a kernel dump to the primary or secondary dump device. When the dump completes, the system halts. Use the **crash** command to examine a kernel dump. Use the **sysdumpdev** command to reassign the dump device.

During a kernel dump, the following values can be displayed on the three-digit terminal display as follows:

0c0 Indicates that the dump completed successfully.
0c1 Indicates that an I/O occurred during the dump. This value only applies to AIX 4.2.1 or later.
0c2 Indicates that the dump is in progress.
0c4 Indicates that the dump is too small.
0c5 Indicates a dump internal error .
0c6 Prompts you to make the secondary dump device ready. This value does not apply for AIX 4.2.1 or later.
0c7 Indicates that the dump process is waiting for a response from the remote host.
0c8 Indicates that the dump was disabled. In this case, no dump device was designated in the system configuration object for dump devices. The **sysdumpstart** command halts, and the system continues running.
0c9 Indicates that a dump is in progress.
0cc Indicates that the system switched to the secondary dump device after attempting a dump to the primary device. This value only applies to AIX 4.2.1 or later.

You can use the Web-based System Manager Devices application (**wsm devices** fast path) to run this command. You could also use the System Management Interface Tool (SMIT) **smit sysdumpstart** fast path to run this command.

Flags

-f Suppresses the prompt to make the secondary dump device ready. This flag does not apply to AIX 4.2.1 or later.
-p Initiates a system dump and writes the results to the primary dump device.
-s Initiates a system dump and writes the results to the secondary dump device.

Security

Access Control: Only the root user can run this command.

Examples

1. To start a kernel dump to the primary dump device, enter:
   ```
   sysdumpstart -p
   ```
2. To start a kernel dump to the secondary dump device, enter:
   ```
   sysdumpstart -s
   ```

Related Information

System Dump Facility in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

The **sysdumpdev** command.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.3 Web-based System Manager Administration Guide*.

sysline Command

Purpose

Displays system status on the status line of a terminal.

Syntax

/usr/bin/sysline [**-b**] [**-c**] [**-d**] [**-e**] [**-h**] [**-i**] [**-j**] [**-l**] [**-m**] [**-p**] [**-q**] [**-r**] [**-s**] [**-w**] [**-D**] [**-H** *Remote*] [**+N**]

Description

The **sysline** command runs in the background and periodically displays system status information on the status line of the terminal. Not all terminals contain a status line. If no flags are specified, the **sysline** command displays the following status items:

- Time of day
- Current number of processes which may be run
- Number of users (followed by a u)
- Number of executable processes (followed by an r)
- Number of suspended processes (followed by an s)
- Number of users who have logged on and off since the last status report

Finally, if new mail has arrived, a summary of it is printed. If there is unread mail in your mailbox, an asterisk appears after the display of the number of users. The display is normally in reverse video (if your terminal supports this in the status line) and is right-justified to reduce distraction. Every fifth display is done in normal video to give the screen a chance to rest.

If you have a file named **.who** in your home directory, then the contents of that file is printed first. One common use of this feature is to alias the **chdir**, **pushd**, and **popd** commands to place the current directory stack in **/.who** after it changes the new directory.

If you have a file named **.syslinelock** in your home directory, then the **sysline** command will not update its statistics and write on your screen, it will just go to sleep for a minute. This is useful if you want to momentarily disable **sysline**. Note that it may take a few seconds from the time the lock file is created until you are guaranteed that **sysline** will not write on the screen.

Flags

-b	Beeps once every half hour and twice every hour.
-c	Clears the status line for five seconds before each redisplay.
-D	Prints out the current day/date before the time.
-d	Prints status line data in human readable format, debug mode.
-e	Prints out only the information. Suppresses the control commands necessary to put the information on the bottom line. This option is useful for putting the output of the **sysline** command onto the mode line of an **emacs** window.
-H *Remote*	Prints the load average on the remote host *Remote.* If the host is down, or is not sending *rwhod* packets, then the down time is printed instead. If the prefix **ucb** is present, then it is removed.
-h	Prints out the host machine's name after the time.
-i	Prints out the process ID of the **sysline** command process onto standard output upon startup. With this information you can send the alarm signal to the **sysline** process to cause it to update immediately. The **sysline** command writes to the standard error, so you can redirect the standard output into a file to catch the process ID.
-j	Left-justifies the **sysline** command output on terminals capable of cursor movement on the status line.

-l	Suppresses the printing of names of people who log in and out.
-m	Suppresses mail check.
+N	Updates the status line every *N* seconds. The default is 60 seconds.
-p	Suppresses the report of the number of processes that are executable and suspended.
-q	Suppresses the printout diagnostic messages if something goes wrong when starting up.
-r	Suppresses reverse video display.
-s	Prints the short form of a line by left-justifying **iff** (if and only if) escapes are not allowed in the status line. Some terminals (the Televideos and Freedom 100 for example) do not allow cursor movements (or other ″intelligent″ operations) in the status line. For these terminals, the **sysline** command normally uses blanks to cause right-justification. This flag disables the adding of blanks.
-w	Prints the status on the current line of the terminal, suitable for use inside a one line window (Window mode).

Examples

To display the day and date, the number of processes which may be run, the number of users, and to clear the screen five seconds before it updates, enter:

```
sysline -Dcr
```

Note: This will only work on screens which have status line capabilities.

Files

/etc/utmp	Contains the names of users who are logged in.
/dev/kmem	Contains the process table.
/var/spool/rwho/whod.*	Contains who/Uptime information for remote hosts.
${HOME}/.who	Specifies information to print on the bottom line.
${HOME}/.syslinelock	Specifies that when it exists, **sysline** does not print.

Related Information

The **pstat** command, **vmstat** command.

---

syslogd Daemon

Purpose

Logs system messages.

Syntax

syslogd [**-d**] [**-s**] [**-f** *ConfigurationFile*] [**-m** *MarkInterval*] [**-r**] [**-R**] [**-n**] [**-N**] [**-p** *LogName*] [**-M all**]

Description

The **syslogd** daemon reads a datagram socket and sends each message line to a destination described by the **/etc/syslog.conf** configuration file. The **syslogd** daemon reads the configuration file when it is activated and when it receives a hangup signal.

The **syslogd** daemon creates the **/etc/syslog.pid** file, which contains a single line with the command process ID used to end or reconfigure the **syslogd** daemon.

A terminate signal sent to the **syslogd** daemon ends the daemon. The **syslogd** daemon logs the end-signal information and terminates immediately.

Each message is one line. A message can contain a priority code, marked by a digit enclosed in < > (angle braces) at the beginning of the line. Messages longer than 900 bytes may be truncated.

The **/usr/include/sys/syslog.h** include file defines the facility and priority codes used by the configuration file. Locally written applications use the definitions contained in the **syslog.h** file to log messages via the **syslogd** daemon.

Flags

-d	Turns on debugging.
-f *ConfigurationFile*	Specifies an alternate configuration file.
-m *MarkInterval*	Specifies the number of minutes between the **mark** command messages. If you do not use this flag, the **mark** command sends a message with **LOG_INFO** priority sent every 20 minutes. This facility is not enabled by a `selector` field containing an ***** (asterisk), which selects all other facilities.
-M all	Specifies not to suppress duplicate messages in logfile. This flag is valid only if used with the **all** argument.
-s	Specifies to forward a ″shortened″ message to another system (if it is configured to do so) for all the forwarding syslog messages generated on the local system.
-r	Suppresses logging of messages received from remote hosts.
-R	When specified, disables the facility to receive messages from the network using the internet domain socket.
-n	Supresses the ″Message forwarded from <log_host_name>: ″ string added to the beginning of the **syslog** message that is forwarded to a remote log host.
-N	Suppresses logging of priority and facility information for each log message.
-p	Specifies an alternate path name for the UNIX datagram socket.

Configuration File

The configuration file informs the **syslogd** daemon where to send a system message, depending on the message's priority level and the facility that generated it.

If you do not use the **-f** flag, the **syslogd** daemon reads the default configuration file, the **/etc/syslog.conf** file.

The **syslogd** daemon ignores blank lines and lines beginning with a # (pound sign).

Format

Lines in the configuration file for the **syslogd** daemon contain a `selector` field, an `action` field, and an optional `rotation` field, separated by one or more tabs.

The `selector` field names a facility and a priority level. Separate facility names with a , (comma). Separate the facility and priority-level portions of the `selector` field with a . (period). Separate multiple entries in the same selector field with a ; (semicolon). To select all facilities, use an * (asterisk).

The `action` field identifies a destination (file, host, or user) to receive the messages. If routed to a remote host, the remote system will handle the message as indicated in its own configuration file. To display messages on a user's terminal, the `destination` field must contain the name of a valid, logged-in system user.

The `rotation` field identifies how rotation is used. If the `action` field is a file, then rotation can be based on size or time, or both. One can also compress and/or archive the rotated files.

Facilities

Use the following system facility names in the `selector` field:

Facility	Description
kern	Kernel
user	User level
mail	Mail subsystem
daemon	System daemons
auth	Security or authorization
syslog	**syslogd** daemon
lpr	Line-printer subsystem
news	News subsystem
uucp	uucp subsystem
local0 through **local7**	Local use
*	All facilities

Priority Levels

Use the following message priority levels in the `selector` field. Messages of the specified priority level and all levels above it are sent as directed.

Priority Level	Description
emerg	Specifies emergency messages (**LOG_EMERG**). These messages are not distributed to all users. **LOG_EMERG** priority messages can be logged into a separate file for reviewing.
alert	Specifies important messages (**LOG_ALERT**), such as a serious hardware error. These messages are distributed to all users.
crit	Specifies critical messages not classified as errors (**LOG_CRIT**), such as improper login attempts. **LOG_CRIT** and higher-priority messages are sent to the system console.
err	Specifies messages that represent error conditions (**LOG_ERR**), such as an unsuccessful disk write.
warning	Specifies messages for abnormal, but recoverable, conditions (**LOG_WARNING**).
notice	Specifies important informational messages (**LOG_NOTICE**). Messages without a priority designation are mapped into this priority message.
info	Specifies informational messages (**LOG_INFO**). These messages can be discarded, but are useful in analyzing the system.
debug	Specifies debugging messages (**LOG_DEBUG**). These messages may be discarded.
none	Excludes the selected facility. This priority level is useful only if preceded by an entry with an * (asterisk) in the same `selector` field.

Destinations

Use the following message destinations in the `action` field.

Destination	Description
File Name	Full path name of a file opened in append mode
@*Host*	Host name, preceded by **@** (at sign)
User[**,** *User*][...]	User names
*	All users

Rotation

Use the following rotation keywords in the `rotation` field.

Keyword	Description
rotate	This keyword must be specified after the `action` field.
size	This keyword specifies that rotation is based on size. It is followed by a number and either a **k** (kilobytes) or **m**(megabytes).

Keyword	Description
time	This keyword specifies that rotation is based on time. It is followed by a number and either a **h**(hour) or **d**(day) or **w**(week) or **m**(month) or **y**(year).
files	This keyword specifies the total number of rotated files. It is followed by a number. If not specified, then there are unlimited number of rotated files.
compress	This keyword specifies that the saved rotated files will be compressed.
archive	This keyword specifies that the saved rotated files will be copied to a directory. It is followed by the directory name.

Examples

1. To log all mail facility messages at the debug level or above to the file **/tmp/mailsyslog**, type:

   ```
   mail.debug /tmp/mailsyslog
   ```

2. To send all system messages except those from the mail facility to a host named `rigil`, type:

   ```
   *.debug;mail.none @rigil
   ```

3. To send messages at the **emerg** priority level from all facilities, and messages at the **crit** priority level and above from the mail and daemon facilities, to users `nick` and `jam`, typer:

   ```
   *.emerg;mail,daemon.crit nick, jam
   ```

4. To send all mail facility messages to all users' terminal screens, type:

   ```
   mail.debug *
   ```

5. To log all facility messages at the debug level or above to the file **/tmp/syslog.out**, and have the file rotated when it gets larger then 500 kilobytes or if a week passes, limit the number of rotated files to 10, use compression and also use **/syslogfiles** as the archive directory, type:

   ```
   *.debug /tmp/syslog.out rotate size 500k time 1w files 10 compress archive /syslogfiles
   ```

Files

/etc/syslog.conf	Controls the output of **syslogd**.
/etc/syslog.pid	Contains the process ID.

Related Information

The **syslog** subroutine.

---

tab Command

Purpose

Changes spaces into tabs.

Syntax

tab [**-e**] [*File ...*]

Description

The **tab** command reads the file specified by the *File* parameter or standard input, and replaces spaces in the input with tab characters wherever the **tab** command can eliminate one or more spaces. If you specify a file with the *File* parameter, the **tab** command writes the resulting file back to the original file. If the input is standard input, the **tab** command writes to standard output. The **tab** command assumes that tab stops are set every eight columns, starting with column nine. The file name specified for the *File* parameter cannot exceed **PATH_MAX**-9 bytes in length.

Flag

-e Replaces only those spaces at the beginning of a line up to the first non-space character.

Example

To replace space characters in the `File` file with tab characters, enter:

```
tab File
```

File

/usr/bin/tab Contains the **tab** command.

Related Information

The **expand** command, **newform** command, **unexpand** command, **untab** command.

Files in *Operating system and device management*.

Input and output redirection in *Operating system and device management*.

---

tabs Command

Purpose

Sets tab stops on terminals.

Syntax

tabs [*TabSpec* ...] [**+m** [*Number*]] [**-T***Terminal ...*]

Description

The **tabs** command specifies tab stops on terminals that support remotely settable hardware tab characters. Tab stops are set according to the *TabSpec* parameter, and previous settings are erased.

When you use the **tabs** command, always refer to the leftmost column number as 1, even if your workstation refers to it as 0.

If you do not specify the *TabSpec* parameter, the default value is **-8**.

The following preset formats can be specified for the *TabSpec* parameter:

-a Sets the tabs to 1, 10, 16, 36, and 72 (IBM System/370™ Assembler first format).

-a2 Sets the tabs to 1, 10, 16, 40, and 72 (IBM System/370 Assembler second format).

-c Sets the tabs to 1, 8, 12, 16, 20, and 55 (COBOL normal format).

-c2 Sets the tabs to 1, 6, 10, 14, and 49 (COBOL compact format, columns 1-6 omitted). With this code, the first column position corresponds to card column 7. One space gets you to column 8, and a tab gets you to column 12. Files using this code should include a format specification of:

```
<:t-c2 m6 s66 d:>
```

-c3 Sets the tabs to 1, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62, and 67 (COBOL compact format with more tabs than **-c2**). These tabs provide the recommended format for COBOL. Files using this code should include a format specification of:

```
<:t-c3 m6 s66 d:>
```

-f Sets the tabs to 1, 7, 11, 15, 19, and 23 (FORTRAN).

-p Sets the tabs to 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, and 61 (PL/I).

-s Sets the tabs to 1, 10, and 55 (SNOBOL).
-u Sets the tabs to 1, 12, 20, and 44.

In addition to the preset formats, the *TabSpec*s parameter can include:

-*Number*
Sets regularly repeating tabs at every *Number* column. (The standard operating system tab setting is **-8**. The **-8** setting is required when using the **nroff** command with the **-h** flag.) Another special case is the **-0** setting, which implies no tabs at all. If more than 20 tabs are set, you must run the **tabs** command twice to clear them.

Number [,*Number*] ...
Sets tabs at the specified column numbers (a comma-separated list in ascending order). You can specify up to 40 numbers. If any number except the first has a plus-sign prefix, the prefixed number is added to the previous number for the next setting. Thus, the tab list specified by **1,10,20,30** provides the same tab settings as the tab list specified by **1,10,+10,+10**.

-*Filep*
Reads the first line of the *Filep* file for a format specification. If the **tabs** command finds a format specification, the **tabs** command sets tabs as specified. If the **tabs** command does not find a format specification, it sets tabs to the system default (**-8**).

It is sometimes convenient to maintain text files with nonstandard tab stop settings (tab stops that are not set at every eighth column). Such files must be converted to a standard format. This is often done by replacing all tab characters with the appropriate number of space characters, before they can be processed by any commands. A format specification occurring in the first line of a text file specifies how tab characters are to be expanded in the remainder of the file.

A format specification consists of a sequence of parameters separated by blanks and surrounded by **<:** and **:>**. Each parameter consists of a letter key, possibly followed immediately by a value. The following parameters are recognized:

ttabs Specifies the tab stop settings for a file. The value of *tabs* must be one of the following:

- A list of column numbers separated by commas, indicating tab stops set at the specified columns.
- A - (dash) followed immediately by an integer *n*, indicating tab stops set at intervals of *n* columns, that is, at 1+*n*, 1+2**n*, and so on.
- A - (dash) followed by the name of a preset tab stop specification.

  Up to 40 numbers are allowed in a comma-separated list of tab stop settings. If any number (except the first one) is preceded by a plus sign, it is taken as an increment to be added to the previous value. Therefore, the formats **t1, 10, 20, 30** and **t1, 10, +10, +10** are considered identical.

  Standard tab stops are specified by **t-8**, or, equivalently, **t1, 9, 17, 25**. This is the tab stop setting that most system utilities assume, and is the most likely setting to find at a terminal. The specification **t-0** specifies no tab stops at all.

  The preset tab stop specifications that are recognized are as follow:

  a 1, 10, 16, 36, 72

  Assembler, IBM System/370, first format

  a2 1, 10, 16, 40, 72

  Assembler, IBM System/370, second format

  c 1, 8, 12, 16, 20, 55

  COBOL, normal format

  c2 1, 6, 10, 14, 49

  COBOL compact format (columns 1-6 omitted). Using this code, the first typed character corresponds to card column 7; one space gets you to column 8; and a tab gets you to column 12. Files using this tab stop setup should include a format specification as follows:

  ```
  <:t-c2 m6 s66 d:>
  ```

  c3 1, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62, 67

  COBOL compact format (columns 1-6 omitted) with more tab stops than **c2**. This is the recommended format for COBOL. The appropriate format specification is:

  ```
  <:t-c3 m6 s66 d:>
  ```

  f 1, 7, 11, 15, 19, 23

  FORTRAN

  p 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61

  PL/I

  s 1, 10, 55

  SNOBOL

  u 1, 12, 20, 44

  UNIVAC 1100 Assembler

ssize Specifies a maximum line size. The value of *size* must be an integer. Size checking is performed after tab characters have been expanded, but before the margin is adjusted.

mmargin Specifies the number of space characters to be added to the beginning of each line. The value of *margin* must be an integer.

d Indicates that the line containing the format specification is to be deleted from the converted file. The *d* parameter takes no value.

e Indicates that the current format is valid only until another format specification is encountered in the file. The *e* parameter takes no value.

Default values, which are assumed for parameters not supplied, are **t-8** and **m0**. If the *s* parameter is not specified, no size checking is performed. If the first line of a file does not contain a format specification, the above defaults are assumed for the entire file. The following is an example of a line containing a format specification:

```
<:t5,10,15 s72:>
```

If a format specification can be disguised as a comment, it is not necessary to code the *d* parameter.

Flags

-T*Terminal* Identifies the terminal so the **tabs** command can set tabs and margins correctly. The *Terminal* variable is one of the terminals specified in the **greek** command. Supported values for the *Terminal* variable include:

ANSI Any ANSI terminal, such as a VT100 terminal.

hp Hewlett-Packard hardcopy terminals.

2621 Hewlett-Packard 2621.

2640 Hewlett-Packard 2640.

2645 Hewlett-Packard 2645.

Additional hardcopy terminals supported by the **tabs** command include:

- 1620
- 1620-12
- 1620-12-8
- 1700
- 1700-12
- 1700-12-8
- 300
- 300-12
- 300s
- 300s-12
- 40-2
- 4000a
- 4000a-12
- 43
- 450
- 450-12
- 450-12-8
- tn1200
- tn300
- oki

If you do not provide the **-T** flag, the value of the environment variable **TERM** is used. If the **-T** flag is provided with no value or if **-T** and **TERM** have invalid values, the error message `unknown terminal` is displayed and the command terminates.

+m *Number* Moves all tabs to the right the number of columns specified by the *Number* variable. This flag also sets the left margin to the column specified by the *Number* variable. If **m** is specified without a value, the default value for the Number variable is 10. The leftmost margin on most workstations is defined by **+m0**. The first column for tabs is defined as column 0 not column 1.

Note: If the same flag occurs more than once, only the last flag takes effect.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

1. To set tabs every four spaces, enter:
   ```
   tabs -4
   ```
2. To set tabs every ten spaces on a VT100 terminal, enter:
   ```
   tabs -10 -TANSI
   ```

File

/usr/bin/tabs Contains the **tabs** command.

Related Information

The **greek** command, **nroff** command, **troff** command.

---

tail Command

Purpose

Displays the last few lines of a file.

Syntax

Standard Syntax

tail [**-f**] [**-c** *Number* | **-n** *Number* | **-m** *Number* | **-b** *Number* | **-k** *Number*] [*File*]

To Display Lines in Reverse Order

tail [**-r**] [**-n** *Number*] [*File*]

Description

The **tail** command writes the file specified by the *File* parameter to standard output beginning at a specified point. If no file is specified, standard input is used. The *Number* variable specifies how many units to write to standard output. The value for the *Number* variable can be a positive or negative integer. If the value is preceded by + (plus sign), the file is written to standard output starting at the specified number of units from the beginning of the file. If the value is preceded by - (minus sign), the file is written to standard output starting at the specified number of units from the end of the file. If the value is not preceded by + (plus sign) or - (minus sign), the file is read starting at the specified number of units from the end of the file.

The type of unit used by the *Number* variable to determine the starting point for the count is determined by the **-b**, **-c**, **-k**, **-m**, or **-n** flag. If one of these flags is not specified, the **tail** command reads the last ten lines of the specified file and writes them to standard output. This is the same as entering **-n 10** at the command line.

The **-m** flag provides consistent results in both single- and double-byte character environments. The **-c** flag should be used with caution when the input is a text file containing multibyte characters, because output can be produced that does not start on a character boundary.

Flags

-b *Number* Reads the specified file beginning at the 512-byte block location indicated by the *Number* variable.

-c *Number* Reads the specified file beginning at the byte location indicated by the *Number* variable.

-f If the input file is a regular file or if the *File* parameter specifies a FIFO (first-in-first-out), the **tail** command does not terminate after the last specified unit of the input file has been copied, but continues to read and copy additional units from the input file as they become available. If no *File* parameter is specified and standard input is a pipe, the **-f** flag is ignored. The **tail -f** command can be used to monitor the growth of a file being written by another process.

-k *Number* Reads the specified file beginning at the 1KB block location indicated by the *Number* variable.

-m *Number* Reads the specified file beginning at the multibyte character location indicated by the *Number* variable. Using this flag provides consistent results in both single- and double-byte character-code-set environments.

-n *Number* Reads the specified file from the first or last line location as indicated by the sign (+ or - or none) of the *Number* variable and offset by the number of lines *Number*.

-r Displays the output from the end of the file in reverse order. The default for the **-r** flag prints the entire file in reverse order. If the file is larger than 20,480 bytes, the **-r** flag displays only the last 20,480 bytes.

The **-r** flag is valid only with the **-n** flag. Otherwise, it is ignored.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To display the last 10 lines of the `notes` file, enter:

   ```
   tail notes
   ```

2. To specify the number of lines to start reading from the end of the `notes` file, enter:

   ```
   tail  -n 20 notes
   ```

3. To display the `notes` file a page at a time, beginning with the 200th byte, enter:

   ```
   tail  -c +200 notes | pg
   ```

4. To follow the growth of a file, enter:

   ```
   tail  -f accounts
   ```

   This displays the last 10 lines of the `accounts` file. The **tail** command continues to display lines as they are added to the `accounts` file. The display continues until you press the Ctrl-C key sequence to stop it.

File

/usr/bin/tail Contains the **tail** command.

Related Information

The **dd** command, **head** command, **more** command, **pg** command.

Files in *Operating system and device management* describes files, file types, and how to name files.

Input and output redirection in *Operating system and device management* describes how the operating system processes input and output.

talk Command

Purpose

Converse with another user.

Syntax

talk {*User* | *User*@*Host* | *Host*!*User* | *Host*.*User* | *Host*:*User* } [*Tty*] [*Pty*]

Description

The **/usr/bin/talk** command allows two users on the same host or on different hosts to have an interactive conversation. The **talk** command opens both a send window and a receive window on each user's display. Each user is then able to type into the send window while the **talk** command displays what the other user is typing.

To initiate a conversation, a local user executes the **talk** command and specifies a remote user's login ID. The remote user's login ID can contain NLS characters. If the remote user is on a remote host, the name of the host must also be specified in one of the following ways:

User@*Host*
Host!*User*
Host.*User*
Host:*User*

When using full domain names, the only valid form for specifying the user and host is *User*@*Host*. For example, `michael@host17.dev.ibm.com` initiates a conversation with user `michael` at host `host17` in the `dev.ibm.com` domain.

When the local user initiates the conversation, a message is sent to the remote user, inviting a conversation. If the local user also specifies `tty`, the invitation message is sent only to the specified terminal. Otherwise, the invitation is sent to the remote user's login terminal. This usually is the console, but it may be another terminal. Once this invitation is received, the **talk** command displays two windows on the local user's terminal and displays progress messages until the remote user responds to the invitation.

> **Note:** If the remote user is running AIXwindows and has no other terminals open, the **talk** command cannot send an invitation.

To have the conversation, the remote user also has to execute the **talk** command from any terminal and specify the local user's account name and host name, if appropriate. When the remote user accepts the invitation, the **talk** command displays two windows on each user's terminal. One window displays what is typed by the local user; the other window displays what is typed by the remote user. To end the conversation, either user can press the Interrupt (Ctrl-C) key sequence and the connection is closed. The Interrupt key sequence can be displayed and modified using the **stty** command.

If the users involved in the conversation are using National Language Support (NLS) capabilities, their terminals must support the printing of NLS characters. The same is true for conversations using Kanji capabilities; the terminals being used must support the printing of Kanji characters.

The **talk** command requires a valid address to which to bind. The host name of the remote machine must be bound to a working network interface, which is usable by other network commands, such as the **ping** command. If a machine has no network interface, that is a standalone machine, it must bind its host name

to the loopback address (127.0.0.1) in order for the **talk** command to work. For example, two users named `local` and `remote` on a standalone machine could initiate a conversation, using the **talk** command, by entering:

```
talk remote@loopback
```

To which user `remote` responds:

```
talk local@loopback
```

To disallow **talk** command invitations, the remote user can issue the **mesg** command.

> **Note:** The **talk** command uses the Talk 4.3 protocol, which is not compatible with AIX 4.2 versions of the **talk** command.

Examples

1. To talk to a user logged in on a remote host, enter:

   ```
   talk dale@host2
   ```

   In this example, the local user wants to talk with user `dale` who is logged in on `host2`.
2. To talk to a user only if that user is logged in on the console of a remote host, enter:

   ```
   talk dale@host2 console
   ```

   User `dale` receives this message only if logged in on the console at `host2`.

Related Information

The **mesg** command, **stty** command.

The **named** daemon, **talkd** daemon.

Communications and networks in *Operating system and device management*.

Conversing with a remote user in *Networks and communication management*.

---

talkd Daemon

Purpose

Provides the server function for the **talk** command.

Syntax

/usr/sbin/talkd [**-s**]

Description

> **Note:** The **talkd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/talkd** daemon is the server that notifies a user (the recipient) that another user (the caller) wants to initiate a conversation. The daemon sets up the conversation if the recipient accepts the invitation. The caller initiates the conversation by executing the **talk** command specifying the recipient. The recipient accepts the invitation by executing the **talk** command specifying the caller.

The **talkd** daemon listens at the socket defined in the **/etc/services** file. When the **talkd** daemon receives a LOOK_UP request from a local or remote **talk** process, the **talkd** daemon scans its internal invitation table for an entry that pairs the client process (the local or remote **talk** process) with a caller.

If no entry exists in the invitation table, the **talkd** daemon assumes that the client process is the caller. The **talkd** daemon then receives the client process' ANNOUNCE request. The **talkd** daemon broadcasts an invitation on the remote computer where the recipient first logged in (unless the caller specifies a particular tty device). This terminal usually is the console, but it may be another terminal.

Otherwise, the invitation is sent to the terminal that the second user first logged in to. This usually is the console, but it may be another terminal.

If an entry does exist in the **talkd** daemon's internal invitation table, the **talkd** daemon assumes that the client is the recipient. The **talkd** daemon returns the appropriate rendezvous address to the **talk** process for the recipient. The recipient process then establishes a stream connection with the caller process.

> **Note:** The **talkd** daemon uses the Talk 4.3 protocol, which is not compatible with AIX 4.2 versions of the **talk** process. The subserver name for the AIX 4.3 protocol is **ntalk**.

Changes to the **talkd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or /**etc**/**services** file. Entering `talkd` at the command line is not recommended. The **talkd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the /**etc**/**inetd.conf** file and the /**etc**/**services** file.

After changing the **/etc/inetd.conf** or /**etc**/**services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

Debugging messages are sent to the **syslogd** daemon.

> **Note:** The **talkd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file.

Manipulating the talkd Daemon with the System Resource Controller

The **talkd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **talkd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled by default in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

-s Turns on socket-level debugging.

Examples

1. To start the **talkd** daemon, enter the following:

   ```
   startsrc  -t ntalk
   ```

   This command starts the **talkd** subserver.
2. To stop the **talkd** daemon normally, enter the following:

   ```
   stopsrc  -t ntalk
   ```

This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.

3. To force stop the **talkd** daemon and all **talkd** connections, enter the following:

   ```
   stopsrc -t  -f ntalk
   ```

   This command terminates all pending connections and existing connections immediately.

4. To display a short status report about the **talkd** daemon, enter the following:

   ```
   lssrc -t ntalk
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

Files

/etc/utmp	Contains data about users currently logged in.

Related Information

The **kill** command, **lssrc** command, **refresh** command, **startsrc** command, **stopsrc** command, **talk** command.

The **inetd** daemon, **syslogd** daemon.

The **/etc/inetd.conf** file format.

TCP/IP daemons in *Networks and communication management*.

---

tapechk Command

Purpose

Performs consistency checking on the streaming tape device.

Syntax

tapechk [**-?**] *Number1 Number2*

Description

The **tapechk** command performs rudimentary consistency checking on an attached streaming tape device. Some hardware malfunctions of a streaming tape drive can be detected by simply reading a tape. The **tapechk** command provides a way to perform tape reads at the file level.

Because the streaming tape drive cannot backspace over physical data blocks or files, the **tapechk** command rewinds the tape to its starting position prior to each check. This command either checks data for the next number of files specified by the *Number1* parameter or skips the next number of files specified by the *Number2* parameter. If you do not specify any parameters, the **tapechk** command rewinds the tape and checks only the first physical block.

The **tapechk** command uses the device in the **TAPE** environment variable if it is defined. Otherwise, the default tape device is **/dev/rmt0**.

Note: The **backup** command allows you to archive files selectively or as an entire file system. It writes data as a continuous stream terminated by a file mark, regardless of the number of files specified. The **tapechk** command perceives each stream of data as a single file, which is important when you specify numeric parameters.

Although you can use the **tapechk** command on any streaming tape cartridge, it is primarily designed for checking tapes written by the **backup** command.

Flag

-? Explains the format of the **tapechk** command.

Note: If you specify the **-?** flag, it must be specified before the *Number1* and *Number2* parameters.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Example

To check the first three files on a streaming tape device, enter:

```
tapechk 3
```

File

/usr/sbin/tapechk Contains the **tapechk** command.

Related Information

The **backup** command.

The **rmt** special file.

Tape drives in *Operating system and device management*.

---

tar Command

Purpose

Manipulates archives.

Syntax

X/Open Standards:

tar {**-c**|**-r**|**-t**|**-u**|**-x**} [**-B**] [**-d**] [**-E**] [**-F**] [**-h**] [**-i**] [**-l**] [**-m**] [**-o**] [**-p**] [**-s**] [**-U**] [**-v**] [**-w**]
[*-Number*] [**-f** *Archive*] [**-b** *Blocks*]
[**-S** [*Feet*] [*Feet @Density*] [*Blocksb*]] [**-L** *InputList*] [**-X** *ExcludeList*]
[**-N** *Blocks*] [**-R**] [**-D**] [**-C** *Directory*] *File* | *Directory* ...

Berkeley Standards:

tar {**c** |**r**|**t**|**u**|**x**} [**bBdDEfFhiILXmNopRsSUvw**[*0-9*]]
[*Blocks*] [*Archive*] [*InputList*] [*ExcludeFile*]
[[*Feet*] | [*Feet@Density*] | [*Blocksb*]] *Directory* | *File* ...

Description

Attention: Because of limitations on header block space in the **tar** command, user numbers (UIDs), and group identification numbers (GIDs) larger than 65,535 will be corrupted when restored to certain systems. The size constraint affects only the ownership and permissions causing no damage to the data. Corruption of the ownership occurs on the following systems:

- Those that do not use `uname` and `gname` fields to check ownership.
- Those that do not have the same user and group IDs as the archiving system.

Notes:

1. The **ustar** header format allows for file sizes to be as large as 8 GB. Therefore, the **tar** command is enabled to archive files of up to 8 GB in size.
2. **tar** does not preserve the sparse nature of any file that is sparsely allocated. Any file that was originally sparse before the restoration will have all space allocated within the filesystem for the size of the file.

The **tar** command manipulates archives by writing files to, or retrieving files from an archive storage medium. The files used by the **tar** command are represented by the *File* parameter. If the *File* parameter refers to a directory, then that directory and recursively all files and directories within it are referenced as well.

The **tar** command looks for archives on the default device (usually tape), unless you specify another device with the **-f** *Archive* flag. When specifying path names that are greater than 100 characters for the United States Tape Archiver (USTAR) format, remember that the path name is composed of a prefix buffer, a / (slash), and a name buffer.

The prefix buffer can be a maximum of 155 bytes and the name buffer can hold a maximum of 100 bytes. If the path name cannot be split into these two parts by a slash, it cannot be archived. This limitation is due to the structure of the **tar** archive headers, and must be maintained for compliance with standards and backwards compatibility. In addition, the length of a destination for a hard or symbolic link (the 'link name') cannot exceed 100 bytes.

When writing to an archive, the **tar** command uses a temporary file (the **/tmp/tar***** file) and maintains in memory a table of files with several links. You receive an error message if the **tar** command cannot create the temporary file, or if there is not enough memory available to hold the link tables.

Two groups of flags exist for the **tar** command: the required flags and the optional flags. The required flags control the actions of the **tar** command and include the **-c**, **-r**, **-t**, **-u**, and **-x** flags. At least one required flag must be selected for the **tar** command to function. Having selected a required flag, you can select an optional flag but none are necessary to control the **tar** command.

Notes:

1. When the storage device is an ordinary file or a block special file, the **-u** and **-r** flags backspace. However, raw magnetic tape devices do not support backspacing. So when the storage device is a raw magnetic tape, the **-u** and **-r** flags rewind the tape, open it, and then read it again.
2. Records are one block long on block magnetic tape, but they are typically less than half as dense on raw magnetic tape. As a result, although a blocked raw tape must be read twice, the total amount of tape motion is less than when reading one-block records from a block magnetic tape once.
3. The structure of a streaming tape device does not support the addition of information at the end of a tape. Consequently when the storage device is a streaming tape, the **-u** and **-r** flags are not valid options. An attempt to use these flags results in the following error message:

```
tar: Update and Replace options not valid for a
streaming tape drive.
```

4. No recovery exists from tape errors.
5. The performance of the **tar** command to the IBM 9348 Magnetic Tape Unit Model 12 can be improved by changing the default block size. To change the block size, enter the following at the command line:

   ```
   chdev -l <device_name> -a block_size=32k
   ```

For more information on using tape devices see the **rmt** special file.

Flags

Flags for the **tar** command are in two groups, the required and the optional. You must supply at least one required flag to control the **tar** command.

Required Flags

-c Creates a new archive and writes the files specified by one or more *File* parameters to the beginning of the archive.

-r Writes the files specified by one or more *File* parameters to the end of the archive. This flag is not valid for any tape devices because such devices do not support the addition of information at the end of a tape.

-t Lists the files in the order in which they appear in the archive. Files can be listed more than once.

-u Adds the files specified by one or more *File* parameters to the end of the archive only if the files are not in the archive already, or if they have been modified since being written to the archive. The **-u** flag is not valid for any tape devices because such devices do not support the addition of information at the end of a tape.

-U Allows archival and extraction of Extended Attributes. The Extended Attributes include Access control list (ACL) also.

-x Extracts the files specified by one or more *File* parameters from the archive. If the *File* parameter refers to a directory, the **tar** command recursively extracts that directory from the archive. If you do not specify the *File* parameter, the **tar** command extracts all of the files from the archive. When an archive contains multiple copies of the same file, the last copy extracted overwrites all previously extracted copies. If the file being extracted does not already exist on the system, the file is created. If you have the proper permissions, the **tar** command restores all files and directories with the same owner and group IDs as they have on the tape. If you do not have the proper permissions, the files and directories are restored with your owner and group IDs. It is not possible to ask for any occurrence of a file other than the last.

Optional Flags

-B Forces input and output blocking to 20 blocks per record. With this option, the **tar** command can work across communications channels where blocking may not be maintained.

-b *Blocks* Specifies the number of 512 bytes blocks per record. Both the default and the maximum is 20, which is appropriate for tape records. Due to the size of interrecord gaps, tapes written with large blocking factors can hold much more data than tapes with only one block per record.

The block size is determined automatically when tapes are read (the **-x** or **-t** function flags). When archives are updated with the **-u** and **-r** functions, the existing record size is used. The **tar** command writes archives using the specified value of the *Blocks* parameter only when creating new archives with the **-c** flag.

For output to ordinary files with the **-f** flag, you can save disk space by using a blocking factor that matches the size of disk blocks (for example, the **-b4** flag for 2048-byte disk blocks).

-C *Directory* Causes the **tar** command to perform a **chdir** subroutine to the directory specified by the *Directory* variable. Using the **-C** flag allows multiple directories not related by a close common parent to be archived, using short relative path names. For example, to archive files from the **/usr/include** and **/etc** directories, you might use the following command:

```
tar  c -C /usr/include File1 File2 -C /etc File3 File4
```

The **-C***Directory* flag must appear after all other flags and can appear in the list of file names given.

-D Suppress recursive processing when directories are specified.

-d Makes separate entries for block files, special character files, and first-in-first-out (FIFO) piped processes. Normally, the **tar** command will not archive these special files. When writing to an archive with the **-d** flag, the **tar** command makes it possible to restore empty directories, special files, and first-in-first-out (FIFO) piped processes with the **-x** flag. **Note:** Although anyone can archive special files, only a user with root user authority can extract them from an archive (FIFO can also be extracted by non-root users).

-E Avoids truncation of the long user and group names during addition of files to new or existing archive.

-F Checks the file type before archiving. Source Code Control Systems (SCCS), Revision Control Systems (RCS), files named **core**, **errs**, **a.out**, and files ending in **.o** (dot o) are not archived.

-f *Archive* Uses the *Archive* variable as the archive to be read or written. When this flag is not specified, the **tar** command uses a system-dependent default file name of the form **/dev/rmt0**. If the *Archive* variable specified is - (minus sign), the **tar** command writes to standard output or reads from standard input. If you write to standard output, the **-c** flag must be used.

-h Forces the **tar** command to follow symbolic links as if they were normal files or directories. Normally, the **tar** command does not follow symbolic links.

-i Ignores header checksum errors. The **tar** command writes a file header containing a checksum for each file in the archive. When this flag is not specified, the system verifies the contents of the header blocks by recomputing the checksum and stops with a directory checksum error when a mismatch occurs. When this flag is specified, the **tar** command logs the error and then scans forward until it finds a valid header block. This permits restoring files from later volumes of a multi-volume archive without reading earlier volumes.

-L *InputList* The *Inputlist* argument to the **-L** option should always be the name of the file that lists the files and directories that need to be archived or extracted.

-l Writes an error message to standard output for each file with a link count greater than 1 whose corresponding links were not also archived. For example, if **file1** and **file2** are hard-linked together and only **file1** is placed on the archive, then the **-l** flag will issue an error message. Error messages are not displayed if the **-l** flag is not specified.

-m Uses the time of extraction as the modification time. The default is to preserve the modification time of the files.

-N *Blocks* Allows the **tar** command to use very large clusters of blocks when it deals with streaming tape archives. Note however, that on input, the **tar** command cannot automatically determine the block size of tapes with very long block sizes created with this flag. In the absence of a **-N** *Blocks* flag, the largest block size that the **tar** command can automatically determine is 20 blocks.

-o Provides backwards compatibility with older versions (non-AIX) of the **tar** command. When this flag is used for reading, it causes the extracted file to take on the User and Group ID (UID and GID) of the user running the program, rather than those on the archive. This is the default behavior for the ordinary user.

-p Restores fields to their original modes, ignoring the present umask. The **setuid**, **setgid**, and tacky bit permissions are also restored to the user with root user authority. This flag restores files to their original mode but does not restore directories to their original mode.

-R Use recursion when directories are specified. Ignored when used with the**-D** option.

-s Tries to create a symbolic link If the **tar** command is unsuccessful in its attempt to link (regular link) two files with the **-s** flag.

-S*Blocks***b**, **-S** *Feet*, **-S** *Feet***@***Density*

Specifies the number of 512KB blocks per volume (first format), independent of the tape blocking factor. You can also specify the size of the tape in feet by using the second form, in which case the **tar** command assumes a default *Density* variable. The third form allows you to specify both tape length and density. Feet are assumed to be 11 inches long to be conservative. This flag lets you deal more easily with multivolume tape archives, where the **tar** command must be able to determine how many blocks fit on each volume.

Notes:

1. Tape drives vary in density capabilities. The *Density* variable calculates the amount of data a system can fit on a tape.
2. When using 1/4-inch tape devices, be sure to take into account the number of tracks on the tape device when specifying the value for the *Feet* variable. For example, a 4-track,1/4-inch tape drive with a 600-foot tape and a density of 8000 bpi can be specified using the **-S** *Feet***@***Density* flag as follows:

   ```
   -S 2400@8000
   ```

   where 600 feet multiplied by 4 tracks equals 2400 feet.

-U

Archives or restores named extended attributes and ACLs. When listing, this option will display the names of any named extended attributes and the type of any ACLs associated with each file that are part of the archive image.

-v

Lists the name of each file as it is processed. With the **-t** flag, **-v** gives more information about the tape entries, including file sizes, times of last modification, User Number (UID), Group Number (GID), and permissions.

-w

Displays the action to be taken, followed by the file name, and then waits for user confirmation. If the response is affirmative, the action is performed. If the response is not affirmative, the file is ignored.

-*Number*

Uses the **/dev/rmt***Number* file instead of the default. For example, the **-2** flag is the same as the **-f/dev/rmt2** file.

-X *ExcludeList*

Excludes the file names or directories given in the *ExcludeList* from the tar archive being created, extracted or listed. The *ExcludeList* shall contain only one filename or directory per line which are to be excluded from the tar archive being created, extracted from or listed. The **-X** option can be specified multiple times and it takes precedence over all other options.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

1. To write the `file1` and `file2` files to a new archive on the default tape drive, enter:

   ```
   tar  -c file1 file2
   ```

2. To extract all files in the `/tmp` directory from the archive file on the `/dev/rmt2` tape device and use the time of extraction as the modification time, enter:

   ```
   tar -xm -f/dev/rmt2 /tmp
   ```

3. To create a new archive file that contains the `file1` file and pass the archive file to the **dd** command to be written to the `/dev/rmt1` device, enter:

   ```
   tar -cvf - file1 | dd of=/dev/rmt1 conv=sync
   ```

4. To display the names of the files in the `out.tar` disk archive file on the current directory, enter:

   ```
   tar -vtf out.tar
   ```

5. To expand the compressed **tar** archive file, `fil.tar.z`, pass the file to the **tar** command, and extract all files from the expanded **tar** archive file, enter:

   ```
   zcat fil.tar.Z | tar -xvf -
   ```

6. To archive the contents of `/usr/include` and `/usr/bin` files using short relative path names, enter:

   ```
   cd /usr
   tar -cvf/dev/rmt0 -C./include . -C ../bin .
   ```

   Note: When specifying multiple instances of the **-C** flag with relative path names, the user must take the previous **-C** flag request into account.

7. To archive to an 8-mm device when using the **-S** flag, enter:

   ```
   tar -cvf /dev/rmt0 -S 4800000b /usr
   ```

   Note: When archiving to an 8-mm device, the **-S** *Feet* and **-S** *Feet@Density* flags are not recommended, since the 8-mm device does not use the concept of density when writing to a tape.

8. To archive a list of all `C` files that is listed in the file through the *InputList* argument of the **-L** option, enter:

   ```
   tar -cvf fl.tar -L fl_list
   ```

   Where `fl_list` is a file consisting a list of all `.c` files in it. This can be obtained as follows:

   ```
   ls *.c > fl_list
   ```

9. To archive a list of all `C` files by setting a variable using the **-L** option, enter:

   ```
   ls *.c > fl_list
   fl=fl_list
   tar -cvf var.tar -L $fl
   ```

10. To avoid the truncation of long user or group names during creation of the archive, enter:

    ```
    tar -cvEf file.tar file
    ```

11. To create a new archive file that contains the `file1` file with ACL and EA, enter:

    ```
    tar -cvUf /tmp/tar.ar file1
    ```

Berkeley Options

The following are examples of the Berkeley options using the **tar** command:

Note: With Berkeley options the arguments to the flags should be given in exact order in which the flags are given below. For example:

```
tar cvfbL test.tar 20 infile
```

where `test.tar` is archive tar file, `20` is number of blocks, and `infile` is *Inputlist* for the archive.

1. To archive all directories and complete filenames listed in input list file **infile** into **ar.tar**, enter :

   ```
   tar cvfL ar.tar infile
   ```

   Where **infile** contains the pathnames of files that are to be archived.

2. To archive files within directories listed in the input list file **infile** into **ar.tar**, enter:

   ```
   tar cvRfL ar.tar infile
   ```

3. To extract directories and complete files specified in the input list file **infile** from an archive named **ar.tar**, enter:

   ```
   tar xvfL ar.tar infile
   ```

4. To extract files from within directories and complete files specified in the input list file **infile** from an archive named **ar.tar**, enter:

   ```
   tar xvRfL ar.tar infile
   ```

Files

/dev/rmt0	Specifies the default tape device.
/bin/tar	Specifies the symbolic link to the **tar** command.
/usr/bin/tar	Contains the **tar** command.
/tmp/tar*	Specifies a temporary file.

Note: In AIX 3.2, the entire **/bin** directory is a symbolic link to **/usr/bin**.

Related Information

The **cat** command, **dd** command, **pax** command.

The **rmt** special file.

File systems in *Operating system and device management* explains file system types, management, structure, and maintenance.

Directory Overview in *Operating system and device management* explains working with directories and path names.

Files in *Operating system and device management* provides information on working with files.

---

tbl Command

Purpose

Formats tables for the **nroff** and **troff** commands.

Syntax

tbl [**-TX**] [**—**] [*File*... | **-**]

Description

The **tbl** command is a preprocessor that formats tables for the **nroff** and **troff** commands. It reads one or more files. If no *File* parameter or **-** (minus sign) is specified as the last parameter, the command reads standard input by default. It copies the input unchanged to standard output, except for text between lines containing **.TS** and **.TE**. The **tbl** command reformats such text, which describes tables, without altering the **.TS** and **.TE** lines.

Depending on the target output device, the output formatted by the **nroff** command may need to be post-processed by the **col** command to produce correct output.

Note: To minimize the volume of data passed through pipelines, enter the **tbl** command first when using it with the **eqn** or **neqn** command.

Input Format

The **tbl** command processes text that is displayed within the following format:

```
[ {.DS .DF} ]
.TS
Options ;
Format .
Data
.TE
[.DE]
```

To include short tables in an **mm** macro document, enclose them within the **.DS** (or **.DF**) and **.DE** macro pair.

Options

Following are the available global options for the input format:

Option	Purpose
center or **CENTER**	Centers the line.
expand or **EXPAND**	Expands to line length.
box or **BOX**	Encloses in a box.
allbox or **ALLBOX**	Boxes all entries.
doublebox or **DOUBLEBOX**	Encloses in two boxes.
tab(*Character***)** or **TAB(***Character***)**	Changes the tab character to the *Character* value.
linesize(*Number***)** or **LINESIZE(***Number***)**	Makes all lines the thickness of the point size specified by the *Number* value.
delim(*XY***)** or **DELIM(***XY***)**	Recognizes the *X* and *Y* variables as **eqn** command delimiters.
;	Denotes end of options.

Format

The *Format* variable in the Input Format describes the format of text. Each format line (the last of which must end with a period) describes all remaining lines of the table. A single-key letter describes each column of each line of the table. Follow this key letter with specifiers that determine the font and point size of the corresponding item, indicate where vertical bars are to displayed between columns, and determine such things as column width and intercolumn spacing. The following are the available key letters:

l or **L** Left-adjusts column.

r or **R** Right-adjusts column.

c or **C** Centers column.

n or **N** Numerically aligns column.
Note: Numerically aligned data, **n** or **N** format specification, are based upon the locale that is specific for *RADIXCHAR*, which is assumed to be a single character. The alignment can also be determined using the \& (backslash, ampersand) character sequence independent of the presence of any *RADIXCHAR* characters. If more than one *RADIXCHAR* character is displayed in a numerically aligned field, the last one is used for alignment. If no *RADIXCHAR* characters are displayed in a particular column, the alignment is based on the last ASCII arabic numeral. If there is no ASCII numeral and no *RADIXCHAR* character in a column, the data is centered.

a or **A** Left-adjusts subcolumn.

s or **S** Spans item horizontally.

t or **T** Pushes vertical spans to top.

v or **V** Adjusts vertical line spacing.

^ Spans item vertically.

u or **U** Moves item half-line up.

z or **Z** Indicates zero-width item.

- Indicates horizontal line.

= Indicates double horizontal line.

| Indicates vertical line.

|| Indicates double vertical line.

b or **B** Indicates boldface item.

i or **I** Indicates italic item.

f*Character* or **F***Character*
Changes to the font specified by the *Character* variable.

p*Number* or **P***Number*
Changes to the size specified by the *Number* variable.

w(*Number***)** or **W(***Number***)**
Sets minimum column width equal to the *Number* variable value.

*Number**Number*
Spaces between columns.

e or **E**
Makes equal-width columns.

.
Ends format.

Data

Handling data within the input format, especially for tables, uses the following line commands:

T{**...T**}
Indicates text block, as follows:

Data<TAB>**T**{

Text Block

T}<TAB>*Data*

_
Writes short horizontal line.

\R*X*
Repeats the *X* parameter value across a column.

\^
Indicates that the item listed previously spans downward into this row.

.T&
Starts new format.

.TS H, **.TH**, and **.TE**
Allows multi-page tables with column headings repeated on each page. (This is a feature of the **mm** macros.)

Parameters

File Specifies the files that the **tbl** command will be processing.

Flags

-TX Uses only full vertical line motions, making the output suitable for line printers and other devices that do not have partial vertical line motions.

— (double dash) Indicates the end of flags.

- Forces input to be read from standard input.

Examples

The following example shows coded input, and associated table output of the **tbl** command. The @ (at sign) is used in input to represent an input tab character.

Input

```
.TS
center box ;
cB s s
cI | cI s
^ | c c
l | n n .
Household Population

_
Town@Households
@Number@Size
=
Bedminster@789@3.26
Bernards Twp.@3087@3.74
```

```
Bernardsville@2018@3.30
Bound Brook@3425@3.04
Bridgewater@7897@3.81
Far Hills@240@3.19
.TE
```

Related Information

The **col** command, **eqn** command, **mm** command, **mmt** command, **mvt** command, **neqn** command, **nroff** command, **soelim** command, **troff** command.

The **mm** macro package, **mv** macro package.

---

tc Command

Purpose

Interprets text into the **troff** command output for the Tektronix 4015 system.

Syntax

tc [**-t**] [**-e**] [**-a** *Number*] [**-o** *List* | **-s** *Number*] [**—**] [*File* | **-**]

Description

The **tc** command interprets input as output from the **troff** command. The **tc** command reads one or more English-language files. If no file is specified or the **-** (minus sign) flag is specified as the last parameter, standard input is read by default. The standard output of the **tc** command is intended for a Tektronix 4015 (a 4014 terminal with ASCII and APL character sets). The various typesetter sizes are mapped into the 4014's four sizes. The entire **troff** command character set is drawn using the 4014 character generator, with overstruck combinations where necessary.

At the end of each page, the **tc** command waits for a new-line character from the keyboard before continuing to the next page. While it waits, the following commands are recognized:

!*Command*	Sends the value of the *Command* variable to the shell.
-e	Does not erase before each page.
-*Number*	Skips backward the specified number of pages.
-a*Number*	Sets the aspect ratio to the value of the *Number* variable.
?	Prints a list of available options.

Note: The **tc** command does not distinguish among fonts.

Parameters

File	Specifies the English-language text files to be interpreted as output from the **troff** command.

Flags

-a *Number*	Sets the aspect ratio to the specified number. The default is 1.5.
-e	Does not erase before each page.
-o *List*	Prints only the pages enumerated in the *List* variable. The list consists of pages and page ranges (for example, 5-17) separated by commas. The range *Number***-** goes from the *Number* variable value to end; the range **-***Number* goes from the beginning to and including the page specified by the *Number* variable.
-s *Number*	Skips the first specified number of pages.

-t Does not wait between pages when directing output into a file.
- Reads from standard input.
— (double dash) Indicates the end of flags.

Example

To use the **tc** command in a pipeline with the **troff** command, enter:

```
troff [Flag...] [File...] | tc
```

Related Information

The **nroff** command, **troff** command.

---

tcbck Command

Purpose

Audits the security state of the system.

Syntax

Check Mode

tcbck { **-n** | **-p** | **-t**| **-y** } [**-i**] [**-o**] { **ALL** | **tree** | { *Name* ... *Class* ... } }

Update Mode

tcbck -a -f *File* | *PathName Attribute* **=** *Value ...*

OR

tcbck -d -f*File* | { *PathName* ... | *Class* ... }

OR

tcbck -l /dev/*filename* /dev/*filename*

Description

The **tcbck** command audits the security state of the system by checking the installation of the files defined in the **/etc/security/sysck.cfg** file (the `sysck` database). Each file definition in the **/etc/security/sysck.cfg** file can include one or more attributes that describe proper installation. When invoked with no flags and with no parameters, the **tcbck** command prints a synopsis of its syntax.

The `tcbck` database usually defines all the files and programs that are part of the trusted computing base, but the root user or a member of the security group can choose to define only those files considered to be security-relevant.

Note: This command writes its messages to **stderr**.

Flags

-a Adds or updates file definitions in the `sysck` database.
-d Deletes file definitions from the `sysck` database.
-f *File* Specifies that file definitions be read from *File*.
-i Excludes filesystems under directories listed in the **treeck_nodir** attribute when the **tree** option is specified.

-l (Lowercase L) Adds entries to the **sysck.cfg** file for **/dev/** files that the administrator would like registered with the Trusted Computing Base.

-n Specifies the checking mode and indicates that errors are to be reported, but not fixed.

-o Writes output to syslog.

-p Specifies the checking mode and indicates that errors are to be fixed, but not reported.

-t Specifies the checking mode and indicates that errors are to be reported with a prompt asking whether the error should be fixed.

-y Specifies the checking mode and indicates that errors are to be fixed and reported.

Modes of Operation

The **tcbck** command has two modes of operation: check mode and update mode. A description of each mode follows.

Check Mode

In check mode, the **tcbck** command checks file definitions against the installed files. You can check all the file definitions in the `sysck` database (the **/etc/security/sysck.cfg** file) by specifying the **ALL** value, or all the files in the file system tree by specifying the **tree** value. If you prefer to check specific files, you can use the *Name* parameter to give the path names of individual files or the *Class* parameter to group several files into a logical group that is defined by a class name, such as audit. You must select one of the following: the **ALL** or **tree** values, or one or more files identified by the *Class* or *Name* parameter.

If the **tree** value is the selection criterion, all the files in the file system tree are checked to ensure that all the relevant files are defined in the `sysck` database. Files defined in the `tcbck` database are checked against their definitions. Files not in the `tcbck` database must *not*:

- Have the **trusted computing base** attribute set.
- Be **setuid** or **setgid** to an administrative ID.
- Be linked to a file in the`tcbck` database.
- Be a device special file.

If the **tcbck** command is running in check mode with both the **tree** value and the **-t** flag and an error occurs, the command provides an error message and prompts you for a decision on how or whether the error should be corrected. If you decide not to delete the file or turn off illegal permissions, you are prompted for a decision on updating the database. If you request an update, the system supplies missing information, such as the name of the file, the link, or the unregistered device name.

A flag (**-n**, **-p**, **-t**, **-y**) also must be included to specify check mode and identify the method of error handling. If there is a duplicate stanza in the **/etc/security/sysck.cfg** file, an error is reported, but not fixed.

Updating the Vital Product Database (VPD) involves defining the **type**, **checksum**, and **size** attributes of each file to the VPD manager. This information is used to verify a correct installation. If these attributes are not defined in **-f** *File*, they are computed when the program is installed or updated. The **checksum** attribute is computed with a method specifically defined for the VPD manager. Refer to “Fixing Errors” on page 328 for more information on file attributes.

The only file definitions modified during an update are the new definitions that indicate a file is part of the trusted computing base (TCB). The *File* parameter is the stanza file that contains the file definitions in **tcbck** format, and is defined in the **/etc/security/sysck.cfg** file. When the update is complete, the files are checked against their file definitions in the stanza file and errors are fixed and reported.

Programs that require **setuid** or **setgid** privilege must be in the `tcbck` database, or these privileges will be cleared when the **tcbck** command runs in Check mode.

Update Mode

In update mode, the **tcbck** command adds (**-a**), deletes (**-d**), or modifies file definitions in the **/etc/security/sysck.cfg** file for the file specified by the *File* parameter, the *PathName* parameter, or the *Class* parameter. The *Class* parameter permits you to group several files into a logical group that is defined by a class name, such as audit. The **tcbck** command also deletes the specified stanzas from the **/etc/security/sysck.cfg** file.

In update mode, the **tcbck** command (**-l**) adds or modifies **/dev/** entry definitions in the **/etc/security/sysck.cfg** file for the specified **/dev** entry. This flag should be run by the administrator to add newly created devices that are trusted to the **sysck.cfg** file. If new devices are not added to the **sysck.cfg** file, the tree option produces warnings of unregistered devices.

The **-l** flag creates a stanza for each **/dev/** entry listed on the command line. The information for the stanza is taken from the current status of the **/dev** entry. The stanza includes:

Device name	**/dev/ entry name**
File type	Either **FILE**, **DIRECTORY**, **FIFO**, **SYMLINK**, **BLK_DEV**, **CHAR_DEV**, or **MPX_DEV**
Owner ID	Owner name
Group ID	Group name
Permissions	Read/write/execute permissions for owner, group and other. **SUID**, **SGID**, **SVTX** and **TCB** attribute bits
Target	If the file is a symbolic link, the target file will be listed.

File definitions to be added or modified with the **-a** flag can be specified on the command line or in a file as *Attribute*=*Value* statements. The following attributes can be used:

acl The access control list for the file. If the value is **blank**, the **acl** attribute is removed. If no value is specified, the command computes a value, according to the format described in Access Control Lists.

class The logical group of the file. A value must be specified, because it cannot be computed. If the value is **blank**, the **class** attribute is removed from the specified file stanza. The value is *ClassName* [*ClassName*].

checksum The checksum of the file. If the value is **blank**, the **checksum** attribute is removed. If no value is specified, the command computes a value, according to the format given in the **sum** command. The value is the output of the **sum -r** command, including spaces.

group The file group. If the value is **blank**, the **group** attribute is removed. If no value is specified, the command computes a value, which can be a group ID or a group name.

links The hard links to this file. If the value is **blank**, the **links** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name, expressed as *Path* [,*Path* ...].

mode The File mode. If the value is **blank**, the **mode** attribute is removed. If no value is specified, the command computes a value, which can be an octal number or string (*rwx*), and have the **tcb**, **SUID**, **SGID**, and **SVTX** attributes.

owner The file owner. If the value is **blank**, the **owner** attribute is removed. If no value is specified, the command computes a value, which can be a user ID or a user name.

program The associated checking program for the file. If the value is **blank**, the **program** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name. If flags are specified, the value should be expressed as *Path*, *Flag*.

symlinks The symbolic links to the file. If the value is **blank**, the **symlinks** attribute is removed. A value must be specified, because it cannot be computed. The value must be an absolute path name, expressed as *Path* [,*Path*..].

size The size of the file in bytes. If the value is **blank**, the **size** attribute is removed. If no value is specified, the command computes a value. The value is a decimal number.

source The source for the file. If the value is **blank**, the **source** attribute is removed. If no value is specified, an empty file of the appropriate type is created. The value must be an absolute path name.

type The type of file. This value cannot be **blank**. If no value is specified, the command computes a value, which can be the **FILE**, **DIRECTORY**, **FIFO**, **BLK_DEV**, **CHAR_DEV**, or **MPX_DEV** keywords.

You can add, delete, or modify the attributes of the **tcbck** command by creating or modifying a **sysck** stanza in the **/etc/security/sysck.cfg** file. The following attributes can be used:

checksum An alternate checksum command to compute the checksum value of files. The system appends the name of each file to the command. If the value is **blank**, this alternate **checksum** attribute is removed. The value is the command string to be run on each file. The default string is **/usr/bin/sum -r <**.

setgids An additional list of administrative groups to be checked for **setgid** programs that are not valid (groups with ID numbers greater than 200). If the value is **blank**, the **setgids** attribute is removed. The value is a comma separated list of group names.

setuids An additional list of administrative users to be checked for **setuid** programs that are not valid (users with ID numbers greater than 200). If the value is **blank**, the **setuids** attribute is removed. The value is a comma separated list of user names.

treeck_nodir A list of directories to be excluded from verification by the **tcbck** command. If the value is blank, the **treeck_nodir** attribute is removed. The value is a comma separated list of directories. File systems that exist under directories contained in this attribute are *not* excluded. Use the **-i** flag to exclude these file systems.

treeck_novfs A list of file systems to be excluded from verification by the **tcbck** command during a check of an installed file system tree. If the value is **blank**, the **treeck_novfs** attribute is removed. The value is a comma separated list of file systems.

Refer to the **/etc/security/sysck.cfg** file for more information about these attributes and “Examples” on page 329 for information about a typical stanza.

If *Attributes* are included without values, the command tries to compute the value from the file to be changed. The **type** attribute is mandatory, but the others do not need to be specified.

Fixing Errors

To fix errors, the **tcbck** command usually resets the attribute to the defined value. For the following attributes, the command modifies its actions as described:

checksum Disables the file by clearing its access control list, but does not stop any further checks.

links Creates any missing hard links. If a link exists to another file, the link is deleted.

program Invokes the program, which must exist and have an absolute path name. A message is printed if an error occurs, but no additional action is taken.

size Disables the file by clearing its access control list, but does not stop any further checks.

source Copies the source file to the file identified by the *File* parameter. If the source is null, any existing file is deleted and a file of the correct **type** is created.

symlinks Creates any missing symbolic links. If a link exists to another file, the link is deleted.

type Disables the file by clearing its access control list, and stops any further checks.

If you used the **-t** flag with the **tcbck** command, you are prompted for a decision on fixing errors. If you answer `yes`, errors are fixed. If you give any other response, errors are not fixed.

Security

Access Control: This command grants execute (x) access only to the root user and members of the security group. The command should be setuid to the root user and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	/etc/passwd

Mode	File
r	/etc/group
r	/etc/security/user
rw	/etc/security/sysck.cfg
x	/usr/bin/aclget
x	/usr/bin/aclput
x	/usr/bin/sum

Auditing Events:

Event	Information
TCBCK_Check	file, error, status
TCBCK_Update	file, function

Examples

1. To add the **/bin/boo** file with **acl**, **checksum**, **class**, **group**, **owner**, and **program** attributes to the `tcbck` database, type:

   ```
   tcbck -a /bin/boo acl checksum class=audit group owner\
   program=/bin/boock
   ```

   The resulting stanza will contain the attributes given previously, with computed values inserted for those attributes you do not define. The database will contain a stanza like the following:

   ```
   /bin/boo:
          acl =
          checksum = 48235
          class = audit
          group = system
          owner = root
          program = /bin/boock
          type = FILE
   ```

   The attribute values are added to the installation definition but are not checked for correctness. The **program** attribute value comes from the command line, the **checksum** attribute value is computed with the **checksum** program, and all the others, except `acl`, are computed from the file i-node.
2. To indicate that the size of a file should be checked but not added to the database, because it can expand during installation, use the **VOLATILE** keyword, as in the following example for the **/etc/passwd** file:

   ```
   /etc/passwd:
           type =  FILE
           owner = root
           group = system
           size  = 1234,VOLATILE
   ```
3. To delete the **/bin/boo** file definition from the `tcbck` database, type:

   ```
   tcbck -d /bin/boo
   ```
4. To delete all definitions with a `class` of `audit` from the `tcbck` database, type:

   ```
   tcbck -d audit
   ```
5. To check all the files in the `tcbck` database, and fix and report all errors, type:

   ```
   tcbck -y ALL
   ```
6. To exclude the `/calvin` and the `/hobbes` file systems from verification during a security audit of an installed file system tree, type:

   ```
   tcbck -a sysck treeck_novfs=/calvin,/hobbes
   ```

7. To exclude a directory from verification during a security audit, type:
   ```
   tcbck -a sysck treeck_nodir=/home/john
   ```
8. To add `jfh` and `jsl` as administrative users and `developers` as an administrative group to be verified during a security audit of an installed file, type:
   ```
   tcbck -a sysck setuids=jfh,jsl setgids=developers
   ```
9. To create/modify **sysck.cfg** stanza entries for the newly created **/dev** entries `foo` and `bar`, type:
   ```
   tcbck -l /dev/foo /dev/bar
   ```

   Note: By adding these entries you are registering them as part of the Trusted computing base.

Attention: Although the special characters ″$″ and ″?″ are allowed in this routine, using them in filenames may result in potential problems such as ambiguous files.

Files

/usr/bin/tcbck	Specifies the path to the **tcbck** command.
/etc/security/sysck.cfg	Specifies the path to the system configuration database.

Related Information

The **aclget** command, **grpck** command, **installp** command, **pwdck** command, **sum** command, **usrck** command.

The Software Vital Product Data (SWVPD) in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

Access control lists in *Operating system and device management*discusses the format of an access control list and provides an example of one.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to Securing the network in *Security*.

---

tcopy Command

Purpose

Copies a magnetic tape.

Syntax

tcopy *Source* [*Destination*]

Description

The **tcopy** command copies magnetic tapes. Source and target file names are specified by the *Source* and *Destination* parameters. The **tcopy** command assumes that there are two tape marks at the end of the tape, and it ends when it finds the double file marks. With only a source tape specified, the **tcopy** command prints information about the size of records and tape files

Examples

To copy from one streaming tape to a 9-track tape, enter:

```
tcopy /dev/rmt0 /dev/rmt8
```

Files

/usr/bin/tcopy Contains the **tcopy** command.

Related Information

Backup files and storage media in *Operating system and device management*.

The **rmt** special file.

---

tcpdump Command

Purpose

Dumps traffic on a network

Syntax

tcpdump [**-a**] [**-A**] [**-d**] [**-D**] [**-e**] [**-f**] [**-I**] [**-L**] [**-n**] [**-N**] [**-O**] [**-p**] [**-q**] [**-R**] [**-S**] [**-t**] [**-u**] [**-U**] [**-v**] [**-x**] [**-X**] [**-c** *count*]
[**-C** *file_size*] [**-F** *file*]
[**-i** *interface*] [**-m** *module*] [**-r** *file*]
[**-s** *snaplen*] [**-w** *file*]
[**-E** *addr*] [**-y** *datalinktype*]
[*expression*]

Description

The **tcpdump** command prints out the headers of packets on a network interface that match the boolean expression. It can also be run with the **-w** flag, which causes it to save the packet data to a file for later analysis. It can also be run with the **-r** flag, which causes it to read from a saved packet file rather than to read packets from a network interface. In all cases, only packets that match expression will be processed by **tcpdump**.

If it is not run with the **-c** flag, **tcpdump** will continue capturing packets until it is interrupted by a SIGINT signal (typically control-C) or a SIGTERM signal (typically the **kill(1)** command). If **tcpdump** is run with the **-c** flag, it will capture packets until it is interrupted by a SIGINT or SIGTERM signal or the specified number of packets have been processed.

When **tcpdump** completes capturing packets, it will report counts of:

packets ″**received by filter**″
Counts all packets regardless of whether they were matched by the filter expression.

packets ″**dropped by kernel**″
The number of packets that were dropped, due to a lack of buffer space.

Allowable Primitives

dst host host
True if the IPv4/v6 destination field of the packet is host, which may be either an address or a name.

src host host
True if the IPv4/v6 source field of the packet is host.

host host
True if either the IPv4/v6 source or destination of the packet is host. Any of the above host expressions can be prepended with the keywords, ip, arp, rarp, or ip6 as in:ip host host which is equivalent to:

```
ether proto \ip and host host
```

If host is a name with multiple IP addresses, each address will be checked for a match.

ether dst ehost
True if the ethernet destination address is `ehost`. Ehost may be either a name from **/etc/ethers** or a number (see ethers(3N) for numeric format).

ether src ehost
True if the ethernet source address is `ehost`.

ether host ehost
True if either the ethernet source or destination address is `ehost`.

gateway host
True if the packet used host as a gateway. For example, the ethernet source or destination address was host but neither the IP source nor the IP destination was host. Host must be a name and must be found both by the machine's host-name-to-IP-address resolution mechanisms (host name file, DNS, NIS, etc.) and by the machine's host-name-to-Ethernet-address resolution mechanism (/etc/ethers, and so on). An equivalent expression is ether host ehost and not host host which can be used with either names or numbers for host /ehost. This syntax does not work in IPv6-enabled configuration at this moment.

dst net net
True if the IPv4/v6 destination address of the packet has a network number of net.

src net net
True if the IPv4/v6 source address of the packet has a network number of net.

net net
True if either the IPv4/v6 source or destination address of the packet has a network number of net.

net net mask netmask
True if the IP address matches net with the specific netmask. May be qualified with src or dst. Note that this syntax is not valid for IPv6 net.

net net/len
True if the IPv4/v6 address matches net with a netmask len bits wide. May be qualified with src or dst.

dst port port
True if the packet is ip/tcp, ip/udp, ip6/tcp orip6/udp and has a destination port value of port. The port can be a number or a name used in **/etc/services** (see tcp(4P) and udp(4P)). If a name is used, both the port number and protocol are checked. If a number or ambiguous name is used, only the port number is checked (e.g., dst port 513 will print both tcp/login traffic and udp/who traffic, and port domain will print both tcp/domain and udp/domain traffic).

src port port
True if the packet has a source port value of port.

port port
True if either the source or destination port of the packet is port. Any of the above port expressions can be prepended with the keywords, tcp or udp, as in: tcp src port port which matches only tcp packets whose source port is port.

less length
True if the packet has a length less than or equal to length. This is equivalent to: len <= length.

greater length
True if the packet has a length greater than or equal to length. This is equivalent to: len >= length.

ip proto protocol
True if the packet is an IP packet of protocol type protocol. Protocol can be a number or one of the names icmp, icmp6, igmp, igrp, pim, ah, esp, vrrp, udp, or tcp. Note that the identifiers tcp, udp, and icmp are also keywords and must be escaped via backslash (\), which is \\ in the C-shell. Note that this primitive does not chase the protocol header chain.

ip6 proto protocol
True if the packet is an IPv6 packet of protocol type protocol. Note that this primitive does not chase the protocol header chain.

ip6 protochain protocol
True if the packet is IPv6 packet, and contains protocol header with type protocol in its protocol header chain. For example, ip6 protochain 6 matches any IPv6 packet with TCP protocol header in the protocol header chain. The packet may contain, for example, authentication header, routing header, or hop-by-hop option header, between IPv6 header and TCP header. The BPF code emitted by this primitive is complex and cannot be optimized by BPF optimizer code in **tcpdump**, so this can be somewhat slow.

ip protochain protocol
Equivalent to ip6 protochain protocol, but this is for IPv4.

ether broadcast
True if the packet is an ethernet broadcast packet. The ether keyword is optional.

ip broadcast
True if the packet is an IPv4 broadcast packet. It checks for both the all-zeroes and all-ones broadcast conventions, and looks up the subnet mask on the interface on which the capture is being done.

If the subnet mask of the interface on which the capture is being done is not available, for example, because the interface on which capture is being done has no netmask this check will not work correctly.

ether multicast
True if the packet is an ethernet multicast packet. The ether keyword is optional. This is shorthand for `ether[0] & 1 != 0`.

ip multicast
True if the packet is an IP multicast packet.

ip6 multicast
True if the packet is an IPv6 multicast packet.

ether proto protocol
True if the packet is of ether type protocol.Protocol can be a number or one of the names ip, ip6, arp, rarp, atalk, aarp, decnet, sca, lat, mopdl, moprc, iso, stp, ipx, or netbeui. Note that these identifiers are also keywords and must be escaped via backslash (\).

[In the case of FDDI (e.g., `fddi protocol arp'), Token Ring (e.g., `tr protocol arp'), and IEEE 802.11 wireless LANS (e.g., `wlan protocol arp'), for most of those protocols, the protocol identification comes from the 802.2 Logical Link Control (LLC) header, which is usually layered on top of the FDDI, Token Ring, or 802.11 header. When filtering for most protocol identifiers on FDDI, Token Ring, or 802.11, tcpdump checks only the protocol ID field of an LLC header in so-called SNAP format with an Organizational UnitIdentifier (OUI) of 0x000000, for encapsulated Ethernet; it doesn't check whether the packet is in SNAP format with an OUI of 0x000000. The exceptions are:

iso **tcpdump** checks the DSAP (Destination Service Access Point) and SSAP (Source Service Access Point) fields of the LLC header.

stp and netbeui
tcpdump checks the DSAP of the LLC header.

atalk **tcpdump** checks for a SNAP-format packet with an OUI of 0x080007 and the AppleTalk etype.

In the case of Ethernet, **tcpdump** checks the Ethernet type field for most of those protocols. The exceptions are:

iso, sap, and netbeui
tcpdump checks for an 802.3 frame and then checks the LLC header as it does for FDDI, Token Ring, and 802.11.

atalk **tcpdump** checks both for the AppleTalk etype in an Ethernet frame and for a SNAP-format packet as it does for FDDI, Token Ring, and 802.11.

aarp **tcpdump** checks for the AppleTalk ARP etype in either an Ethernet frame or an 802.2 SNAP frame with an OUI of 0x000000;

ipx **tcpdump** checks for the IPX etype in an Ethernet frame, the IPX DSAP in the LLC header, the 802.3-with-no-LLC-header encapsulation of IPX, and the IPX etype in a SNAP frame.

decnet src host
True if the DECNET source address is host, which may be an address of the form 10.123, or a DECNET host name. [DECNET host name support is only available on Ultrix systems that are configured to run DECNET.]

decnet dst host
True if the DECNET destination address is host.

decnet host host
True if either the DECNET source or destination address is host.

ifname interface
True if the packet was logged as coming from the specified interface.

on interface
Synonymous with the ifname modifier.

rnr num
True if the packet was logged as matching the specified PF rule number (applies only to packets logged by OpenBSD's pf(4)).

rulenum num
Synonomous with the rnr modifier.

reason code
True if the packet was logged with the specified PF reason code. The known codes are: match, bad-offset, fragment, short, normalize, and memory (applies only to packets logged by OpenBSD's **pf(4)**).

action act
True if PF took the specified action when the packet was logged. Known actions are: pass and block (applies only to packets logged by OpenBSD's **pf(4)**)

netbeui
ip, ip6, arp, rarp, atalk, aarp, decnet, iso, stp, ipx.

Abbreviations for:

```
ether proto p
```

where *p* is one of the above protocols.

lat, moprc, mopdl

Abbreviations for:

```
ether proto p
```

where *p* is one of the above protocols. Note that **tcpdump** does not currently know how to parse these protocols.

vlan [*vlan_id***]**
True if the packet is an IEEE 802.1Q VLAN packet. If [*vlan_id*] is specified, only true is the packet has the specified *vlan_id*. Note that the first **vlan** keyword encountered in expression changes the decoding offsets for the remainder of expression on the assumption that the packet is a VLAN packet.

tcp, udp, icmp
Abbreviations for:

```
ip proto p or ip6 proto p
```

where *p* is one of the above protocols.

iso proto protocol
True if the packet is an OSI packet of protocol type protocol. Protocol can be a number or one of the names clnp, esis, or isis.

clnp, esis, isis
Abbreviations for:

- iso proto p

where *p* is one of the above protocols.

l1, l2, iih, lsp, snp, csnp, psnp
Abbreviations for IS-IS PDU types.

vpi *n* True if the packet is an ATM packet, for SunATM on Solaris, with a virtual path identifier of *n*.

vci *n* True if the packet is an ATM packet, for SunATM on Solaris, with a virtual channel identifier of *n*.

lane True if the packet is an ATM packet, for SunATM on Solaris, and is an ATM LANE packet. Note that the first lane keyword encountered in expression changes the tests done in the remainder of expression on the assumption that the packet is either a LANE emulated Ethernet packet or a LANE LE Control packet. If lane isn't specified, the tests are done under the assumption that the packet is an LLC-encapsulated packet.

llc True if the packet is an ATM packet, for SunATM on Solaris, and is an LLC-encapsulated packet.

oamf4s
True if the packet is an ATM packet, for SunATM on Solaris, and is a segment OAM F4 flow cell (VPI=0 & VCI=3).

oamf4e
True if the packet is an ATM packet, for SunATM on Solaris, and is an end-to-end OAM F4 flow cell (VPI=0 & VCI=4).

oamf4 True if the packet is an ATM packet, for SunATM on Solaris, and is a segment or end-to-end OAM F4 flow cell (VPI=0 & (VCI=3 | VCI=4)).

oam True if the packet is an ATM packet, for SunATM on Solaris, and is a segment or end-to-end OAM F4 flow cell (VPI=0 & (VCI=3 | VCI=4)).

metac True if the packet is an ATM packet, for SunATM on Solaris, and is on a meta signaling circuit (VPI=0 & VCI=1).

bcc True if the packet is an ATM packet, for SunATM on Solaris, and is on a broadcast signaling circuit (VPI=0 & VCI=2).

sc True if the packet is an ATM packet, for SunATM on Solaris, and is on a signaling circuit (VPI=0 & VCI=5).

ilmic True if the packet is an ATM packet, for SunATM on Solaris, and is on an ILMI circuit (VPI=0 & VCI=16).

connectmsg
True if the packet is an ATM packet, for SunATM on Solaris, and is on a signaling circuit and is a Q.2931 Setup, Call Proceeding, Connect, Connect Ack, Release, or Release Done message.

metaconnect
True if the packet is an ATM packet, for SunATM on Solaris, and is on a meta signaling circuit and is a Q.2931 Setup, Call Proceeding, Connect, Release, or Release Done message.

expr relop expr
True if the relation holds, where relop is one of >, <, >=, <=, =, !=, and expr is an arithmetic expression composed of integer constants (expressed in standard C syntax), the normal binary operators [+, -, *, /, &, |], a length operator, and special packet data accessors. To access data inside the packet, use the following syntax:

```
proto [ expr : size ]
```

Proto is one of ether, fddi, tr, wlan, ppp, slip, link, ip, arp, rarp, tcp, udp, icmp or ip6, and indicates the protocol layer for the index operation. (ether, fddi, wlan, tr, ppp, slip and link all refer to the link layer.) Note that tcp, udp and other upper-layer protocol types only apply to IPv4, not IPv6 (this will be fixed in the future). The byte offset, relative to the indicated protocol layer, is given by expr. Size is optional and indicates the number of bytes in the field of interest; it can be either one, two, or four, and defaults to one. The length operator, indicated by the keyword len, gives the length of the packet.

For example, `ether[0] & 1 != 0` catches all multicast traffic. The expression `ip[0] & 0xf !=5` catches all IP packets with options. The expression `ip[6:2] & 0x1fff = 0` catches only unfragmented datagrams and frag zero of fragmented datagrams. This check is implicitly applied to the tcp and udp index operations. For instance, tcp[0] always means the first byte of the TCP header, and never means the first byte of an intervening fragment.

Some offsets and field values may be expressed as names rather than as numeric values. The following protocol header field offsets are available: icmptype (ICMP type field), icmpcode (ICMP code field), and tcpflags (TCP flags field).

The following ICMP type field values are available: icmp-echoreply, icmp-unreach, icmp-sourcequench, icmp-redirect, icmp-echo, icmp-routeradvert, icmp-routersolicit, icmp-timxceed, icmp-paramprob, icmp-tstamp, icmp-tstampreply, icmp-ireq, icmp-ireqreply, icmp-maskreq, icmp-maskreply.

The following TCP flags field values are available: tcp-fin, tcp-syn, tcp-rst, tcp-push, tcp-ack, tcp-urg.

Combining Primitives

A parenthesized group of primitives and operators (parentheses are special to the Shell and must be escaped).

```
Negation (`!' or `not').

Concatenation (`&&' or `and').

Alternation (`||' or `or').
```

Negation has highest precedence. Alternation and concatenation have equal precedence and associate left to right. Note that explicits and tokens, not juxtaposition, are now required for concatenation.

If an identifier is given without a keyword, the most recent keyword is assumed. For example, `not host vs and ace` is short for `not host vs and host ace` which should not be confused with `not ( host vs or ace )`

Expression arguments can be passed to **tcpdump** as either a single argument or as multiple arguments, whichever is more convenient. Generally, if the expression contains Shell metacharacters, it is easier to pass it as a single, quoted argument. Multiple arguments are concatenated with spaces before being parsed.

Flags

-a Attempt to convert network and broadcast addresses to names.

-A Print each packet (minus its link level header) in ASCII. Handy for capturing web pages.

-c Exits after receiving *Count* packets.

-C *file_size* Before writing a raw packet to a *savefile*, check whether the file is currently larger than *file_size* and, if so, close the current savefile and open a new one. Savefiles after the first *savefile* will have the name specified with the **-w** flag, with a number after it, starting at 2 and continuing upward. The units of *file_size* are millions of bytes (1,000,000 bytes, not 1,048,576 bytes).

-d Dumps the compiled packet-matching code to standard output, then stops.

-D Print the list of the network interfaces available on the system and on which **tcpdump** can capture packets. For each network interface, a number and an interface name (possibly followed by a text description of the interface) is printed. The interface name or the number can be supplied to the **-i** flag to specify an interface on which to capture.

-dd Dump packet-matching code as a C program fragment.

-ddd Dump packet-matching code as decimal numbers (preceded with a count).

-e Prints the link-level header on each dump line.

-E *addr* Use `spi@ipaddr algo:secret` for decrypting IPsec ESP packets that are addressed to *addr* and contain Security Parameter Index value `spi`. This combination may be repeated with comma or newline separation.
Note: Setting the secret for IPv4 ESP packets is now supported.

Algorithms may be `des-cbc`, `3des-cbc`, `blowfish-cbc`, `rc3-cbc`, `cast128-cbc`, or none. The default is `des-cbc`. The ability to decrypt packets is only present if **libcrypto** is installed and is in LIBPATH.

`secret` is the ASCII text for ESP secret key. If preceeded by `0x`, then a hex value will be read.

The option assumes RFC2406 ESP, not RFC1827 ESP. The option is for debugging purposes only and the use of this option with a true secret key is discouraged. By presenting the IPsec secret key onto command line you make it visible to others, via **ps(1)** and other occasions.

In addition to the above syntax, the syntax file name may be used to have **tcpdump** read the provided file. The file is opened upon receiving the first ESP packet, so any special permissions that **tcpdump** may have been given, should already have been given up.

-f Prints foreign IPv4 addresses numerically rather than symbolically.

The test for foreign IPv4 addresses is done by using the IPv4 address and netmask of the interface on which capture is being performed. This option will not work correctly if that address or netmask is not available.

-F *file* Use *file* as input for the filter expression. An additional expression given on the command line is ignored.

-i *interface* Listens on *interface*. If unspecified, **tcpdump** searches the system *interface* list for the lowest numbered, configured up *interface* (excluding loopback). Ties are broken by choosing the earliest match.

An *interface* number as printed by **-D** flag can be used as the *interface* argument.

-l Makes *stdout* line buffered. Useful if you want to see the data while capturing it. For example:

```
tcpdump -l | tee dat
  or
tcpdump -l   >   dat & tail -f dat
```

-L Lists the known data link types for the interface and exits.

-m *module* Loads SMI MIB module definitions from the *module* file. This option can be used several times to load several MIB modules into **tcpdump**.

-n Blocks converting addresses (i.e., host addresses, port numbers, etc.) to names.

-N Omits printing domain name qualification of host names. For example,**tcpdump** will print `nic` instead of `nic.ddn.mil`.

-O Keeps **tcpdump** from running the packet-matching code optimizer. This is useful only if you suspect a bug in the optimizer.

-p Won't put the interface into promiscuous mode. Note that the interface might be in promiscuous mode for some other reason; hence, **-p** cannot be used as an abbreviation for `ether host {local-hw-addr}` or `ether broadcast`.

-q Quick output. Prints less protocol information so output lines are shorter.

-r *file* Read packets from *file* (which was created with the **-w** option). Standard input is used if *file* is ″-″.

-R Assumes ESP/AH packets are based on old specification.

(RFC1825 to RFC1829). If specified, **tcpdump** will not print replay prevention field. Since there is no protocol version field in ESP/AH specification, **tcpdump** cannot deduce the version of ESP/AH protocol.

-S Prints absolute rather than relative TCP sequence numbers.

-s *snaplen* Snarf *snaplen* bytes of data from each packet rather than the default of 68. 68 bytes is adequate for IP, ICMP, TCP and UDP but may truncate protocol information from name server and NFS packets (see below). Packets truncated because of a limited snapshot are indicated in the output with [l*proto*], where *proto* is the name of the protocol level at which the truncation has occurred. Note that taking larger snapshots increases the amount of time it takes to process packets and effectively decreases the amount of packet buffering. This can cause packets to be lost. You should limit *snaplen* to the smallest number that will capture the protocol information you are interested in. Setting *snaplen* to 0 means use the required length to catch whole packets.

-T Forces packets selected by *expression* to be interpreted the specified type. Currently known types are cnfp (Cisco NetFlow protocol), rpc (Remote Procedure Call), rtp (Real-Time Applications protocol), rtcp (Real-Time Applications control protocol), snmp (Simple Network Management Protocol), tftp (Trivial File Transfer Protocol), vat (Visual Audio Tool), and wb (distributed White Board).

-t Omits the printing of a timestamp on each dump line.

-tt Prints an unformatted timestamp on each dump line.

-ttt Prints a delta (in micro-seconds) between current and previous line on each dump line.

-tttt Prints a timestamp in default format proceeded by date on each dump line.

-u Prints undecoded NFS handles.

-U Make output saved via the **-w** option, for example, ″packet- buffered.″ As each packet is saved, it will be written to the output file, rather than being written only when the output buffer fills.

-v Specifies slightly more verbose output. For example, the time to live, identification, total length and options in an IP packet are printed. Also enables additional packet integrity checks such as verifying the IP and ICMP header checksum.

-vv Even more verbose output than **-v**. For example, additional fields are printed from NFS and reply packets are fully decoded.

-vvv Even more verbose output than **-vv**. For example, telnet SB ... SE options are printed in full. With **-X** Telnet options are printed in hex as well.

-w *file* Writes the raw packets to *file* rather than parsing and printing them out. They can later be printed with the **-r** flag. Standard output is used if *File* is ″-″.

-x Prints each packet (minus its link level header) in hexadecimal. The smaller of the entire packet or snaplen bytes will be printed. Note that this is the entire link-layer packet, so for link layers that pad (e.g. Ethernet), the padding bytes will also be printed when the higher layer packet is shorter than the required padding.

-xx Prints each packet, including its link level header, in hexadecimal.

-X Prints each packet (minus its link level header) in hexadecimal and ASCII. This is very handy for analyzing new protocols.

-y *datalinktype* Set the data link type to use while capturing packets to *datalinktype*.

Parameters

expressions

Selects which packets will be dumped. If no expression is given, all packets on the net will be dumped. Otherwise, only packets for which expression is `true` will be dumped.

The expression consists of one or more primitives. Primitives usually consist of an id (name or number) preceded by one or more qualifiers. There are three different kinds of qualifier:

type qualifiers say what type of primitive the id name or number refers to. Possible types are host, net and port. For example, `host foo', `net 128.3', `port 20'. If there is no type qualifier, host is assumed.

dir qualifiers specify a particular transfer direction to and/or from id. Possible directions are src, dst, src or dst and src and dst. If there is no dir qualifier, src or dst is assumed. For some link layers, such as SLIP and for some other device types, the inbound and outbound qualifiers can be used to specify a desired direction.

proto qualifiers restrict the match to a particular protocol. Possible protos are fddi, tr, wlan, ip, ip6, arp, rarp, decnet, tcp and udp. If there is no proto qualifier, all protocols consistent with the type are assumed.

fddi is an alias for ether. The parser treats it as meaning ″the data link level used on the specified network interface.″ FDDI headers contain Ethernet-like source and destination addresses, and often contain Ethernet-like packet types, so you can filter on these FDDI fields just as with the analogous Ethernet fields. FDDI headers also contain other fields, but they cannot be named in a filter expression.

Like **fddi**, **tr** and **wlan** are aliases for ether. The previous paragraph's statements about FDDI headers also apply to Token Ring and 802.11 wireless LAN headers. For 802.11 headers, the destination address is the DA field and the source address is the SA field; the BSSID, RA, and TA fields aren't tested.

In addition to the above, there are some special `primitive' keywords that don't follow the pattern: gateway, broadcast, less, greater and arithmetic expressions. All of these are described below.

More complex filter expressions are built by using the words `and`, `or`, and `not` to combine primitives.

Environment Variables

LIBPATH environmental variable must be set or **libcrypto** library should be in **/usr/lib** for the **-E** flag to work. For example:

```
ksh$ LIBPATH=/opt/freeware/lib tcpdump -E"algo:secret"
```

Exit Status

0 Success.

non-zero Error.

Security

Reading packets from a network interface requires read access to **/dev/bpf***, which is typically root-only. Reading packets from a file does not require any special privileges except file read permission.

Examples

1. To print all packets arriving at or departing from sundown, type the following:
   ```
   tcpdump host sundown
   ```
2. To print traffic between helios and either hot or ace, type the following:
   ```
   tcpdump host helios and \( hot or ace \)
   ```
3. To print all IP packets between ace and any host except helios, type the following:
   ```
   tcpdump ip host ace and not helios
   ```
4. To print all traffic between local hosts and hosts at Berkeley, type:
   ```
   tcpdump net ucb-ether
   ```
5. To print all ftp traffic through internet gateway snup, type the following:
   ```
   tcpdump 'gateway snup and (port ftp or ftp-data)'
   ```

   Note: The expression is quoted to prevent the shell from (mis-)interpreting the parentheses)
6. To print traffic neither sourced from nor destined for local hosts (if you gateway to one other net, this should never make it onto your local net), type the following.
   ```
   tcpdump ip and not net localnet
   ```
7. To print the start and end packets (the SYN and FIN packets) of each TCP conversation that involves a non-local host, type:
   ```
   tcpdump 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0 and not src and d dst net localnet'
   ```
8. To print IP packets longer than 576 bytes sent through gateway snup, type the following:
   ```
   tcpdump 'gateway snup and ip[2:2] > 576'
   ```
9. To print IP broadcast or multicast packets that were not sent via ethernet broadcast or multicast, type:
   ```
   tcpdump 'ether[0] & 1 = 0 and ip[16] >= 224'
   ```
10. To print all ICMP packets that are not echo requests/replies (for instance, not ping packets), type:
    ```
    tcpdump 'icmp[icmptype] != icmp-echo and icmp[icmptype] != icmp-e choreply'
    ```

Standard Error

All errors and warnings are sent to **stderr**.

Limitations

A packet trace that crosses a daylight savings time change will give skewed time stamps (the time change is ignored).

Filter expressions on fields other than those in Token Ring headers will not correctly handle source-routed Token Ring packets.

Filter expressions on fields other than those in 802.11 headers will not correctly handle 802.11 data packets with both To DS and From DS set.

ip6 proto should chase header chain, but at this moment it does not. ip6 protochain is supplied for this behavior.

Arithmetic expression against transport layer headers, like tcp[0], does not work against IPv6 packets. It only looks at IPv4 packets.

Files

/usr/sbin/tcpdump	Location of the **tcpdump** command.
/usr/lib/libpcap.a	

/dev/bpf*

/opt/freeware/lib/libcrypto.a(libcrypto.so) Optional

Related Information

pcap library, iptrace, Berkeley Packet Filter.

---

tctl Command

Purpose

Gives subcommands to a streaming tape device.

Syntax

tctl [**-f** *Device*] [**eof** | **weof** | **fsf** | **bsf** | **fsr** | **bsr** | **rewind** | **offline** | **rewoffl** | **erase** | **retension** | **reset** | **status**] [*Count*]

tctl [**-b** *BlockSize*] [**-f** *Device*] [**-p** *BufferSize*] [**-v**] [**-n**] [**-B**] { **read** | **write** }

Description

The **tctl** command gives subcommands to a streaming tape device. If you do not specify the *Device* variable with the **-f** flag, the **TAPE** environment variable is used. If the environment variable does not exist, the **tctl** command uses the **/dev/rmt0.1** device. (When the **tctl** command gives the **status** subcommand, the default device is **/dev/rmt0**.) The *Device* variable must specify a raw (not block) tape device. The *Count* parameter specifies the number of end-of-file markers, number of file marks, or number of records. If the *Count* parameter is not specified, the default count is 1.

Subcommands

eof or **weof**
Writes the number of end-of-file markers specified by the *Count* parameter at the current position on the tape. On an 8 mm tape drive, an end-of-file marker can be written in three places:

- Before blank tape
- Before an extended file mark
- At the beginning-of-tape mark

On a 9-track tape drive, the end-of-tape marker can be written at any location on the tape. However, this subcommand does not support overwriting single blocks of data.

fsf
Moves the tape forward the number of file marks specified by the *Count* parameter and positions it on the end-of-tape (EOT) side of the file mark.

bsf
Moves the tape backward the number of file marks specified by the *Count* parameter and positions it on the beginning-of-tape (BOT) side of the file mark. If the **bsf** subcommand moves the tape past the beginning, the tape rewinds, and the **tctl** command returns **EIO**.

fsr
Moves the tape forward the number of records specified by the *Count* parameter.

bsr
Moves the tape backwards the number of records specified by the *Count* parameter.

rewind
Rewinds the tape. The *Count* parameter is ignored.

offline or **rewoffl**
Rewinds the tape and takes the tape drive offline. This will unload the tape when appropriate. The tape must be re-inserted before the device can be used again.

erase
Erases all contents on the tape and rewinds it.

read	Reads from the specified tape device (using the specified block size) until the internal buffer is full, and then writes the data to standard output, continuing to read and write this way until an end-of-file (EOF) mark is reached.
reset	Sends a bus device reset (BDR) to the tape device. The BDR will only be sent if the device cannot be opened and is not busy.
retension	Moves the tape to the beginning, then to the end, and then back to the beginning of the tape. If you have excessive read errors during a restore operation, you should run the **retension** subcommand. If the tape has been exposed to environmental extremes, you should run the **retension** subcommand before writing to tape. The 8 mm tape drive will not respond to this command.
status	Prints status information about the specified tape device.
write	Opens the tape device, reads from standard input, and writes the data to the tape device.

Notes:

1. When you specify the **read** or **write** subcommand, the **tctl** command opens the tape device and sets up the tape block size as specified by the **-b** or **-n** flag. If neither flag is specified, the **tctl** command uses a default block size of 512 bytes.
2. The **-b**, **-n**, **-p**, and **-v** flags apply only when using the **read** and **write** subcommands.
3. The **-B** flag applies only when using the **read** subcommand.

Flags

-b *BlockSize*	Specifies, in bytes, the size of buffer used to read and write to the tape device, and also specifies, in the absence of the **-n** flag, the tape block size. If the block size is 0, variable-length blocks are used and the size of the tape buffer is 32,768. If the **-b** flag is not specified, the default block size and the size of the tape buffer is 512 bytes.
-B	Writes the contents of the buffer each time the tape is read. Set this flag when reading variable-length records that are not of a regular and consistent size.
-f *Device*	Specifies the tape device.
-p *BufferSize*	Specifies the size of the buffer to be used on standard input and standard output. The default buffer size is 32,768 bytes. The *BufferSize* value must be a multiple of the tape block size.
-v	Verbose. Prints the sizes of each read and write to standard error.
-n	Specifies variable-length records when reading or writing to tape with the **read** or **write** subcommand.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To rewind the rmt1 tape device, enter:

   ```
   tctl  -f /dev/rmt1  rewind
   ```

2. To move forward two file marks on the default tape device, enter:

   ```
   tctl  fsf 2
   ```

3. To write two end-of-file markers on the tape in /dev/rmt0.6, enter:

```
tctl  -f /dev/rmt0.6  weof 2
```

4. To read a tape device formatted in 80-byte blocks and put the result in a file, enter:

```
tctl  -b 80  read > file
```

5. To read variable-length records from a tape device formatted in 80-byte blocks and put the result in a file, enter:

```
tctl  -b 80  -n  read > file
```

6. To write variable-length records to a tape device using a buffer size of 1024 byes, enter:

```
cat file | tctl  -b 1024  -n  -f/dev/rmt1  write
```

7. To write to a tape device in 512-byte blocks and use a 5120-byte buffer for standard input, enter:

```
cat file | tctl  -v  -f /dev/rmt1  -p 5120  -b 512  write
```

   Note: The only valid block sizes for quarter-inch (QIC) tape drives are 0 and 512.

8. To write over one of several backups on an 8 mm tape, position the tape at the start of the backup file and issue these commands:

```
tctl  bsf 1

tctl  eof 1
```

The first command moves the tape to the beginning-of-tape side of the file mark. The second command rewrites the file mark, because writing is allowed before extended file marks. The erase head of the drive erases data before the write head reaches it, so the **write** subroutines can write over data already in the tape. However, all old data following is lost because its file markers are meaningless.

> **Note:** The **write** subroutines cannot write over a short file mark unless blank tape follows the short file mark. To write over existing data, as in the case of this example, the tape must be written with extended file marks (as specified through the SMIT interface).

Files

/dev/rmt*n*	Specifies the raw streaming tape interface.
/usr/bin/tctl	Contains the **tctl** command.

Related Information

The **dd** command, **mt** command.

The **environment** file, **rmt** special file.

The **ioctl** subroutine.

Backup files and storage media in *Operating system and device management*.

---

tee Command

Purpose

Displays the output of a program and copies it into a file.

Syntax

tee [**-a**] [**-i**] [*File* ...]

Description

The **tee** command reads standard input, then writes the output of a program to standard output and simultaneously copies it into the specified file or files.

Flags

-a Adds the output to the end of *File* instead of writing over it.
-i Ignores interrupts.

Exit Status

This command returns the following exit values:

0 The standard input was successfully copied to all output files.
>0 An error occurred.

Note: If a write to any successfully opened *File* operand is not successful, writes to other successfully opened *File* operands and standard output will continue, but the exit value will be **>0**.

Examples

1. To view and save the output from a command at the same time:

   ```
   lint program.c | tee program.lint
   ```

   This displays the standard output of the command **lint program.c** at the workstation, and at the same time saves a copy of it in the file `program.lint`. If a file named `program.lint` already exists, it is deleted and replaced.
2. To view and save the output from a command to an existing file:

   ```
   lint program.c | tee -a program.lint
   ```

   This displays the standard output of the **lint program.c** command at the workstation and at the same time appends a copy of it to the end of the `program.lint` file. If the `program.lint` file does not exist, it is created.

Files

/usr/bin/tee Contains the **tee** command.

Related Information

The **script** command.

Input and output redirection in *Operating system and device management* describes how the operating system processes input and output and how to use the redirect and pipe symbols.

telinit or init Command

Purpose

Initializes and controls processes.

Syntax

{ **telinit** | **init** } { **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **a** | **b** | **c** | **h** | **Q** | **q** | **S** | **s** | **M** | **m** | **N** }

Description

The **init** command initializes and controls processes. Its primary role is to start processes based on records read from the **/etc/inittab** file. The **/etc/inittab** file usually requests that the **init** command run the **getty** command for each line on which a user can log in. The **init** command controls autonomous processes required by the system.

The process that constitutes the majority of the **init** command's process dispatching activities is **/usr/sbin/getty**. The **/usr/sbin/getty** process initiates individual terminal lines. Other processes typically dispatched by the **init** command are daemons and the shell.

The **telinit** command, which is linked to the **init** command, directs the actions of the **init** command. The **telinit** command takes a one-character argument and signals the **init** command by way of the **kill** subroutine to perform the appropriate action.

The **telinit** command sets the system at a specific run level. A run level is a software configuration that allows only a selected group of processes to exist. The system can be at one of the following run levels:

0-9 Tells the **init** command to place the system in one of the run levels **0-9**. When the **init** command requests a change to run levels **0-9**, it kills all processes at the current run levels and then restarts any processes associated with the new run levels.

0-1 Reserved for the future use of the operating system.

2 Contains all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system. The console device driver is also set to run at all run levels so the system can be operated with only the console active.

3-9 Can be defined according to the user's preferences.

S,s,M,m Tells the **init** command to enter the maintenance mode. When the system enters maintenance mode from another run level, only the system console is used as the terminal.

The following arguments also serve as directives to the **init** command:

a,b,c,h Tells the **init** command to process only those records in the **/etc/inittab** file with **a**, **b**, **c**, or **h** in the run level field. These four arguements, **a**, **b**, **c**, and **h**, are not true run levels. They differ from run levels in that the **init** command cannot request the entire system to enter run levels **a, b, c**, or **h**.

When the **init** command finds a record in the **/etc/inittab** file with a value of **a**, **b**, **c**, or **h** in the run level field, it starts the process. However, it does not kill any processes at the current run level; processes with a value of **a**, **b**, **c**, or **h** in the run level field are started in addition to the processes already running at the current system run level. Another difference between true run levels and **a**, **b**, **c**, or **h** is that processes started with **a**, **b**, **c**, or **h** are not stopped when the **init** command changes run levels. Three ways stop **a**, **b**, **c**, or **h** processes:

- Type **off** in the *Action* field.
- Delete the objects entirely.
- Use the **init** command to enter maintenance state.

Q,q Tells the **init** command to re-examine the **/etc/inittab** file.

N Sends a signal that stops processes from being respawned.

During system startup, after the root file system has been mounted in the pre-initialization process, the following sequence of events occurs:

1. The **init** command is run as the last step of the startup process.
2. The **init** command attempts to read the **/etc/inittab** file.
3. If the **/etc/inittab** file exists, the **init** command attempts to locate an `initdefault` entry in the **/etc/inittab** file.
   a. If the `initdefault` entry exists, the **init** command uses the specified run level as the initial system run level.
   b. If the `initdefault` entry does not exist, the **init** command requests that the user enter a run level from the system console (**/dev/console**).
   c. If the user enters an **S**, **s**, **M** or **m** run level, the **init** command enters maintenance run level. These are the only run levels that do not require a properly formatted **/etc/inittab** file.
4. If the **/etc/inittab** file does not exist, the **init** command places the system in the maintenance run level by default.
5. The **init** command rereads the **/etc/inittab** file every 60 seconds. If the **/etc/inittab** file has changed since the last time the **init** command read it, the new commands in the **/etc/inittab** file are executed during system startup.

When you request the **init** command to change the run level, the **init** command reads the **/etc/inittab** file to identify what processes should exist at the new run level. Then, the **init** command cancels all processes that should not be running at the new level and starts any processes that should be running at the new level.

The processes run by the **init** command for each of these run levels are defined in the **/etc/inittab** file. The run level is changed by having a root user run the **telinit** command, which is linked to the **init** command. This user-run **init** command sends appropriate signals to the original **init** command initiated by the system during startup. The default run level can be changed by modifying the run level for the `initdefault` entry in the **/etc/inittab** file.

In the maintenance run level, the **/dev/console** console terminal is opened for reading and writing. The password for root is prompted. When the root password is entered successfully, the **su** command is invoked. Two ways exist to exit from the maintenance run level:

- If the shell is terminated, the **init** command requests a new run level.

  OR
- The **init** (or **telinit**) command can signal the **init** command and force it to change the run level of the system.

During a system startup attempt, apparent failure of the **init** command to prompt for a new run level (when **initdefault** is maintenance) may be due to the fact that the terminal console device (**/dev/console**) has been switched to a device other than the physical console. If this occurs and you wish to work at the physical console rather than the **/dev/console**, you can force the **init** command to switch to the physical console by pressing the DEL (delete) key at the physical console device.

When the **init** command prompts for a new run level, enter one of the digits **0** through **9** or any of the letters **S**, **s**, **M**, or **m**. If you enter **S**, **s**, **M**, or **m**, the **init** command operates in maintenance mode with the additional result that if control had previously been forced to switch to the physical console, the **/dev/console** file is switched to this device as well. The **init** command generates a message to this effect on the device to which the **/dev/console** file was previously connected.

If you enter a **0** through **9** run level, the **init** command enters the corresponding run level. The **init** command rejects any other input and re-prompts you for the correct input. If this is the first time the **init** command enters any run level other than maintenance, it searches the **/etc/inittab** file for entries with the **boot** or **bootwait** keywords. If the **init** command finds these keywords, it performs the corresponding task,

provided the run level entered matches that of the entry. For example, if the **init** command finds the **boot** keyword, it boots the machine. Any special initialization of the system, such as checking and mounting file systems, takes place before any users are allowed on the system. The **init** command then scans the **/etc/inittab** file to find all entries that are processes for that level. It then resumes normal processing of the **/etc/inittab** file.

Run level **2** is defined by default to contain all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system.

For terminal processes, the shell terminates either as a result of an end of file character (EOF) typed explicitly or as the result of disconnection. When the **init** command receives a signal telling it that a process has terminated, it records the fact and the reason it stopped in **/etc/utmp** file and /**var**/**adm**/**wtmp** file. The /**var**/**adm**/**wtmp** file keeps a history of the processes started.

To start each process in the **/etc/inittab** file, the **init** command waits for one of its descendant processes to stop, for a power fail signal **SIGPWR**, or until the **init** command is signaled by the **init** or **telinit** commands to change the system's run level. When one of the above three conditions occurs, the **init** command re-examines the **/etc/inittab** file. Even if new entries have been added to the **/etc/inittab** file, the **init** command still waits for one of the three conditions to occur. To provide for instantaneous response, re-examine the **/etc/inittab** file by running the **telinit -q** command.

If the **init** command finds that it is continuously running an entry in the **/etc/inittab** file (more than five times in 225 seconds), it assumes that an error in the entry command string exists. It then prints an error message to the console and logs an error in the system error log. After the message is sent, the entry does not run for 60 seconds. If the error continues to occur, the command will respawn the entry only five times every 240 seconds. The **init** command continues to assume an error occurred until the command does not respond five times in the interval, or until it receives a signal from a user. The **init** command logs an error for only the first occurrence of the error.

When the **init** command is requested to change run levels by the **telinit** command, the **init** command sends a **SIGTERM** signal to all processes that are undefined in the current run level. The **init** command waits 20 seconds before stopping these processes with the **SIGKILL** signal.

If the **init** command receives a **SIGPWR** signal and is not in maintenance mode, it scans the **/etc/inittab** file for special power fail entries. The **init** command invokes the tasks associated with these entries (if the run levels permit) before any further processing takes place. In this way, the **init** command can perform cleanup and recording functions whenever the system experiences a power failure. It is important to note that these power fail entries should not use devices that need to be initialized first.

Environments

Because the **init** command is the ultimate ancestor of every process on the system, every other process on the system inherits the **init** command's environment variables. As part of its initialization sequence, the **init** command reads the **/etc/environment** file and copies any assignments found in that file into the environment passed to all of its subprocesses. Because **init** subprocesses do not run from within a login session, they do not inherit a umask setting from **init**. These processes may set the umask to whatever value they require. A command that is executed by **init** from the **/etc/inittab** file uses **init**'s ulimit values and not the default values as given in **/etc/security/limits**. The result is that a command that is successfully executed from the command line may not execute correctly when invoked by **init**. Any command that has specific **ulimit** requirements should include specific actions to set the **ulimit** values as required.

Examples

1. To request the **init** command to reexamine the **/etc/inittab** file, enter:

   ```
   telinit  q
   ```

2. To request the **init** command to enter maintenance mode, enter:

   ```
   telinit  s
   ```

Files

/etc/inittab	Specifies the **init** command control file.
/etc/utmp	Specifies the record of logged-in users.
/var/adm/wtmp	Specifies the permanent login accounting file.
/sbin/rc.boot	Specifies the pre-initialization command file.
/etc/rc	Specifies the initialization command file.
/etc/environment	Specifies system environment variables.
/dev/console	Specifies the console device driver.

Related Information

The **chitab** command, **lsitab** command, **mkitab** command, **rmitab** command, **getty** command, **rc** command.

The **inittab** file, the **rc.boot** file.

The **reboot** subroutine, **umask** subroutine, **ulimit** subroutine, **kill** subroutine.

---

telnet, tn, or tn3270 Command

Purpose

Connects the local host with a remote host, using the Telnet interface.

Syntax

{ **telnet** | **tn** | **tn3270** } [**-d**] [**-p**] [**-n** *TraceFile*] [**-e** *TerminalType*] [**-f** | **-F**] [**-k** *realm*] [**-l** user] [*Host* [*Port*]]

Description

The **telnet** command, which is also referred to as the **tn** or **tn3270** command, operates in two different modes: command mode and input mode.

Command Mode

When the **telnet** command is issued without arguments, it enters command mode, as indicated by the `telnet>`, `tn>`, or the `tn3270>` prompt. A user can also enter command mode from input mode by pressing Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. In command mode, subcommands can be entered to manage the remote system. Some of these subcommands return you to the remote session upon completion. For those subcommands that do not, pressing the Enter key returns you to the remote session.

> **Note:** The default escape sequence for this command is Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. This default can be overridden by changing the **TNESC** environment variable.

To enter **telnet** command mode while connected to a remote host, type the Telnet escape key sequence. When in command mode, the standard operating system editing conventions, such as backspace, are available.

Input Mode

When the **telnet** command is issued with arguments, it performs an **open** subcommand with those arguments and then enters input mode. The type of input mode is either character-at-a-time or line-by-line, depending on what the remote system supports. In character-at-a-time mode, most text that is typed is immediately sent to the remote host for processing. In line-by-line mode, all text is echoed locally and completed lines are sent to the remote host.

In either input mode, if the **toggle localchars** subcommand has a value of True, the user's QUIT, INTR, and FLUSH characters are trapped locally and sent as Telnet Protocol sequences to the remote host. The **toggle autoflush** and **toggle autosynch** subcommands cause this action to flush subsequent output to the terminal until the remote host acknowledges the Telnet sequence and to flush previous terminal input (in the case of QUIT and INTR characters).

Arabic/Hebrew Support

The **telnet**, **tn**, and **tn3270** command supports the Arabic and Hebrew texts, allowing the user to type Arabic or Hebrew characters while in an emulation session. The **Ar_AA** locale displays the Arabic characters in their correct shapes. The following functions support the bidirectional Arabic and Hebrew texts:

Language Selection: This function allows you to toggle the language layer. Activate the Arabic/Hebrew language selection with the following key combinations:

Alt+N	From an AIX terminal
Esc+N	From an ASCII terminal
Alt+N or **Esc+N**	From a Latin AIX terminal

Activate the Latin language layer with the following key combinations:

Alt+L	From an Arabic or Hebrew AIX terminal
Esc+L	From an ASCII terminal
Alt+L or **Esc+L**	From an AIX terminal

Screen Reverse: This function reverses the screen image and invokes the default language of the new screen orientation. Thus, if the screen is reversed to right-to-left, the language is changed to Arabic/Hebrew. If the screen is reversed to left-to-right, the language is changed to Latin.

If symmetric character swapping is enabled, reversing the screen causes bidirectional characters to be replaced by their counterparts. For example, if numeric character swapping is enabled, reversing the screen causes Hindi numerals to be replaced by their Arabic counterparts and the Arabic numerals to be replaced by their Hindi counterparts.

Activate screen reverse with the following key combinations:

Alt+S	From an Arabic or Hebrew AIX terminal
Esc+S	From an ASCII terminal
Alt+S or **Esc+S**	From a Latin AIX terminal

Push/End Push: The Push function allows you to edit text whose direction is opposite the screen orientation. When you activate this function, the cursor orientation is reversed, the language layer is changed accordingly, and a Push segment is created.

The Push function has two secondary modes:

Boundary Mode	This mode is activated upon entering the Push mode. In this mode, the cursor remains in its position while you type additional characters. The text is pushed in the opposite direction of the screen orientation.
Edit Mode	This mode is activated when the cursor is moved from its boundary position into the Push segment area. In this mode, you can edit the text within the Push segment, while typing in the field's natural direction.

Activate this function with the following key combinations:

Alt+P	From an Arabic or Hebrew AIX terminal
Esc+P	From an ASCII terminal
Alt+P or **Esc+P**	From a Latin AIX terminal

The End Push function terminates the Push function. The cursor jumps to the end of the Push segment and its direction changes to the original direction. You can activate End Push by pressing any field exit keys such as cursor up, cursor down, or any attention identifier (AID) key such as the Enter key. You can also activate this function with the following key combinations:

Alt+E	From an Arabic or Hebrew AIX terminal
Esc+E	From an ASCII terminal
Alt+E or **Esc+E**	From a Latin AIX terminal

Field Reverse: This function toggles the field orientation to either the opposite of or the same as the screen orientation. This function does not invert the text in the field. The cursor orientation is set to the new field orientation and the language layer is selected accordingly.

For example, if the cursor is in the first logical position of a field or line when you activate the field reverse function, the cursor skips to the opposite side of that field or line. This position is now the first logical position. If the cursor is not in the first position of the field or line when you activate field reverse function, the cursor remains in its position and allows natural and correct editing of the existing text. Activate this function with the following key combinations:

Alt+R	From an Arabic or Hebrew AIX terminal
Esc+R	From an ASCII terminal
Alt+R or **Esc+R**	From a Latin AIX terminal

Autopush: This function assists you in typing mixed left-to-right and right-to-left text. When enabled, reversed segments are automatically initiated and terminated according to the typed characters or the selected language layer. Thus, this mode automatically invokes the Push mode and relieves you of invoking the Push function.

When you type a digit or Latin character in a right-to-left field, the Autopush function automatically initiates the Push function without changing the language. If you type additional digits or Latin character, the Push function continues; otherwise, the Push function automatically terminates. Thus, you can type Arabic/Hebrew text with embedded digits or Latin characters without invoking the Push/End Push functions.

When you type an Arabic/Hebrew character in a left-to-right field, the Autopush function automatically initiates the Push function without a language change. If you then type a digit or Latin character, the Autopush function automatically terminates. Thus, you can type Latin text with embedded Arabic/Hebrew text using the Language Selection function rather than the Push/End Push functions.

Activate this function with the following key combinations:

Alt+A	From an Arabic or Hebrew AIX terminal
Esc+A	From an ASCII terminal
Alt+A or **Esc+A**	From a Latin AIX terminal

Field Shape: This function shapes the Arabic characters in the current field or line. Activate this function with the following key combinations:

Alt+H	From an Arabic AIX terminal
Esc+H	From an ASCII terminal
Alt+H or **Esc+H**	From a Latin AIX terminal

Field Deshape: This function deshapes Arabic text in the current field or line. Activate this function with the following key combinations:

Alt+B	From an Arabic AIX terminal
Esc+B	From an ASCII terminal
Alt+B or **Esc+B**	From a Latin AIX terminal

Contextual Shape Determination: This function determines the shape of an Arabic character based on the surrounding text. Use the Contextual Shape Determination function only when typing or editing right-to-left text. This function is terminated when any of the specific shape selection keys is pressed. This is the default function. Activate this function with the following key combinations:

Alt+C	From an Arabic AIX terminal
Esc+C	From an ASCII terminal
Alt+C or **Esc+C**	From a Latin AIX terminal

Initial Shape Determination: This function shapes Arabic characters in their initial shapes. Activate this function with the following key combinations:

Alt+I	From an Arabic AIX terminal
Esc+I	From an ASCII terminal
Alt+I or **Esc+I**	From a Latin AIX terminal

Middle Shape Determination: This function shapes Arabic characters in their middle shapes. Activate this function with the following key combinations:

Alt+M	From an Arabic AIX terminal
Esc+M	From an ASCII terminal
Alt+M or **Esc+M**	From a Latin AIX terminal

Isolated Shape Determination: This function shapes Arabic characters in their isolated shapes. Activate this function with the following key combinations:

Alt+O	From an Arabic AIX terminal
Esc+O	From an ASCII terminal
Alt+O or **Esc+O**	From a Latin AIX terminal

Final Shape Determination: This function shapes Arabic characters in their final shapes. Activate this function with the following key combinations:

Alt+Y	From an Arabic AIX terminal
Esc+Y	From an ASCII terminal
Alt+Y or **Esc+Y**	From a AIX terminal

Miscellaneous Functions: To activate numeric swapping, type the following line at the command line:

```
export ARB_NUM_SWAP=1
```

To activate symmetric swapping, that is, to swap bidirectional characters such as braces, brackets, and so on, type the following line at the command line:

```
export ARB_SYM_SWAP=1
```

To specify the code page that the host uses, type the following line at the command line:

```
export RM_HOST_LANG=IBM-420
```

Terminal Type Negotiation

The **telnet** command negotiates the terminal type, using the Telnet protocol, and it sets the **TERM** environment variable according to what has been negotiated.

To override the terminal negotiation from the console, use the **EMULATE** environment variable or the **-e** flag; or invoke the tn3270 command if you require 3270 emulation. To determine whether terminal-type negotiation is performed, the following list describes the order of the **telnet** command processing:

1. The **-e** command-line flag. (No negotiation.)
2. The **EMULATE** environment variable. (No negotiation.)
3. The **tn3270** command. (No negotiation.)
4. If steps 1, 2, and 3 are not present, terminal-type negotiation occurs automatically.

If the client and the server negotiate to use a 3270 data stream, the keyboard mapping is determined by the following precedence:

$HOME/.3270keys	Specifies the user's 3270 keyboard mapping when the **tn** or **telnet** command is invoked. If you are using a color display, you can also change this file to customize the colors for 3270 displays.
/etc/map3270	Specifies the user's 3270 keyboard mapping when the **tn3270** command is invoked. The **/etc/map3270** file defines keyboard mapping and colors for the **tn3270** command.
/etc/3270.keys	Specifies the base 3270 keyboard mapping for use with limited function terminals.

Secure Attention Key (SAK) Option

In addition to terminal negotiation, the **telnet** command allows negotiation for the Secure Attention Key (SAK) option. This option, when supported, provides the local user with a secure communication path to the remote host for tasks such as changing user IDs or passwords. If the remote host supports the **SAK** function, a trusted shell is opened on the remote host when the **telnet send sak** subcommand is issued. The **SAK** function can also be assigned to a single key available in **telnet** input mode, using the **set sak** subcommand.

End-of-Line Convention

The Telnet protocol defines the carriage-return line-feed (CR-LF) sequence to mean ″end-of-line.″ For terminal input, this corresponds to a command-completion or end-of-line key being pressed on a user terminal. On an ASCII terminal, this is the CR key, but it may also be labeled ″Return″ or ″Enter.″

When a Telnet server receives the Telnet end-of-line sequence, CR-LF, as input from a remote terminal, the effect is the same as if the user had pressed the end-of-line key on a local terminal.

On ASCII servers, receiving the Telnet sequence CR-LF causes the same effect as a local user pressing the CR key on a local terminal. CR-LF and CR-NUL have the same effect on an ASCII server when received as input over a Telnet connection.

> **Note:** A Telnet user must be able to send CR-LF, CR-NULL, or LF. An ASCII user must be able to send CR-LF or CR-NULL.

A Telnet user on an ASCII host should have a user-controllable mode to send either CR-LF or CR-NULL when the user presses the end-of-line key. The CR-LF should be the default. The Telnet end-of-line sequence, CR-LF, must be used to send Telnet data that is not terminal-to-computer. This occurs, for example, when a Telnet server sends output or when the Telnet protocol incorporates another application protocol.

The **telnet** command ″execs″ (using the **exec** command) the **/usr/sbin/login** command to validate a user. This 1) allows all user and device attributes to take effect on telnet connections and 2) causes telnet connections to count against the maximum number of login sessions allowable at a time (determined by the **maxlogins** attribute). Attributes are defined in the **/etc/security/user** and **/etc/security/login.cfg** files.

Restrictions

- Earlier versions of the **telnet** command are not compatible with AIX Version 4 and later of the **telnet** command in sending escapes that emulate a high function terminal (HFT). AIX Version 4 and later of the **telnet** command sends only one escape when the escape key is hit, while prior versions send two escape characters.
- The **telnet** command must allow transmission of 8-bit characters that are not in binary mode to implement ISO 8859 Latin code page. This is necessary for internationalization of the TCP/IP commands.
- In order to support new character sets, the following was added to the `hft-m`, `ibm5081`, `hft`, `hft-nam`, `hft-c`, `aixterm-m`, and `aixterm` entries in the **terminfo** file:

```
box1=\154\161\153\170\152\155\167\165\166\164\156,     batt1=f1,
box2=\154\161\153\170\152\155\167\165\166\164\156,     batt2=f1md,
font0=\E(B,     font1=\E(0,
```

- The **rlogind** and **telnetd** daemons use POSIX line discipline to change the line discipline on the local tty. If POSIX line discipline is not used on the local tty, echoing other line disciplines may result in improper behavior. AIX TCP/IP must have POSIX line discipline to function properly.
- The mouse cannot be used as an input device with the **telnet** command.
- The **telnet** command does not support the APL data stream.

Environment Variables

The following environment variables can be used with the **telnet** command:

EMULATE Overrides terminal-type negotiation in the same way as the **-e** flag. If the value of the **EMULATE** environment variable is defined as **vt100** or **3270**, the **telnet** command emulates a DEC VT100 terminal or 3270 terminal, respectively. If the **EMULATE** variable is not defined or has a value of **none**, the **telnet** command operates normally. If the **EMULATE** variable is set to **vt100** or **3270**, the **TERM** environment variable in the remote login connection should be set to the same value. You can check this by using the **env** command after the connection is open.

TNESC Specifies an alternate TELNET escape character, other than the default, Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command. To change the **telnet** escape sequence, set **TNESC** to the octal value of the character you want to use. Then export **TNESC**. For example, set **TNESC** to 35 to change the TELNET escape sequence to Ctrl-].

MAP3270 Specifies an alternate file that contains the user's 3270 keyboard mapping. The **MAP3270** variable must contain the full path name to the alternate file. Create the alternate file using the same format as the default **/etc/map3270** file.

RM_HOST_LANG Specifies the EBCDIC code page being used on the remote 3270 host. Set the **RM_HOST_LANG** environment variable to the correct code page before you telnet (using the **telnet** command) to a non-English-speaking 3270 host. The default is English. Refer to the Converters Overview for Programming in *AIX 5L Version 5.3 National Language Support Guide and Reference* for possible code pages to use. Format the **RM_HOST_LANG** environment variable by specifying the desired code page.
Note: The **tn3270** command does not support DBCS, because terminal types for DBCS are not supported.

The **telnet** command converts characters by using the **iconv** command. Users can change the default conversion tables by using the **genxlt** command.

Flags

-d Turns debugging mode on.

-e *TerminalType* Overrides terminal-type negotiation. Possible values are **vt100**, **3270**, or **none**.

-n *TraceFile* Records network trace information in the file specified by the *TraceFile* variable.

-p Preserves current tty attributes.

-f Causes the credentials to be forwarded. This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.

-F Causes the credentials to be forwarded. In addition, the credentials on the remote system will be marked forwardable (allowing them to be passed to another remote system). This flag will be ignored if Kerberos 5 is not the current authentication method. Authentication will fail if the current DCE credentials are not marked forwardable.

-k *realm* Allows the user to specify the realm of the remote station if it is different from the local systems realm. For these purposes, a *realm* is synonymous with a DCE cell. This flag will be ignored if Kerberos 5 is not the current authentication method.

-l *user* Specifies the remote user the telnet wants to login as. This option is ignored if Kerberos 5 is not the current authentication method.

Subcommands

Before entering each subcommand, press the escape key sequence. The escape sequence tells the program that non-text information follows. Otherwise, the program interprets subcommands as text.

For each of the subcommands in the following list, you only need to type enough letters to uniquely identify the subcommand. (For example, **q** is sufficient for the **quit** subcommand.) This is also true for the arguments to the **display**, **emulate**, **mode**, **set**, and **toggle** subcommands.

The **telnet** subcommands are:

? [*Subcommand*] Requests help on **telnet** subcommands. Without arguments, the **?** subcommand prints a help summary. If a *Subcommand* variable is specified, help information is displayed for that subcommand.

close Closes the TELNET connection and returns to **telnet** command mode when the **open** subcommand is used to establish the connection. When the **telnet** command is invoked and a host is specified, the **close** subcommand closes the TELNET connection and exits the **telnet** program (identical to the **quit** subcommand).

display [*Argument*] Displays all of the **set** and **toggle** values if no *Argument* variable is specified; otherwise, lists only those values that match the *Argument* variable.

emulate *TerminalType* Overrides terminal-type negotiation with the specified terminal type. Possible choices are:

? Prints help information.

3270 Emulates a 3270 terminal.

none Specifies no emulation.

vt100 Emulates a DEC VT100 terminal.

All output received from the remote host is processed by the specified emulator. The initial terminal type to emulate can be specified through the **EMULATE** environment variable or the **-e** flag to the **telnet** command.

Note: Only standard ASCII characters are allowed in emulation mode.

mode *Type* Specifies the current input mode. When the *Type* variable has a value of **line**, the mode is line-by-line. When the *Type* variable has a value of **character**, the mode is character-at-a-time. Permission is requested from the remote host before entering the requested mode, and if the remote host supports it, the new mode is entered.

open *Host* [*Port*] Opens a connection to the specified host. The *Host* specification can be either a host name or an Internet address in dotted-decimal form. If no *Port* variable is specified, the **telnet** subcommand attempts to contact a TELNET server at the default port.

quit Closes a TELNET connection and exits the **telnet** program. A Ctrl-D in command mode also closes the connection and exits.

send *Arguments* Sends one or more arguments (special character sequences) to the remote host. Multiple arguments are separated by spaces. The following arguments can be used:

? Prints help information for the **send** subcommand.

ao Sends the TELNET AO (Abort Output) sequence, which causes the remote host to flush all output from the remote system to the local terminal.

ayt Sends the TELNET AYT (Are You There) sequence, to which the remote system can respond.

brk Sends the TELNET BRK (Break) sequence, which causes the remote system to perform a kill operation.

ec Sends the TELNET EC (Erase Character) sequence, which causes the remote host to erase the last character entered.

el Sends the TELNET EL (Erase Line) sequence, which causes the remote system to erase the line currently being entered.

escape
Sends the current **telnet** escape character. The default escape sequence is Ctrl-] for the **telnet** command, Ctrl-T for the **tn** command, or Ctrl-C for the **tn3270** command.

ga Sends the TELNET GA (Go Ahead) sequence, which provides the remote system with a mechanism to signal the local system to return control to the user.

ip Sends the TELNET IP (Interrupt Process) sequence, which causes the remote system to cancel the currently running process.

nop Sends the TELNET NOP (No Operation) sequence.

sak Sends the TELNET SAK (Secure Attention Key) sequence, which causes the remote system to invoke the trusted shell. If the SAK is not supported, then an error message is displayed that reads: `Remote side does not support SAK.`

synch Sends the TELNET SYNC sequence, which causes the remote system to discard all previously typed input that has not yet been read. This sequence is sent as TCP/IP urgent data.

set *VariableValue* Sets the specified TELNET variable to the specified value. The special value **off** turns off the function associated with the variable entered. The **display** subcommand can be used to query the current setting of each variable. The variables that can be specified are:

echo Toggles between local echo of entered characters and suppressing local echo. Local echo is used for normal processing, while suppressing the echo is convenient for entering text that should not be displayed on the screen, such as passwords. This variable can only be used in line-by-line mode.

eof Defines the character for the **telnet** command. When the **telnet** command is in line-by-line mode, entering the eof character as the first character on a line sends the character to the remote host. The initial value for the eof character is the local terminal End-Of-File character.

erase Defines the erase character for the **telnet** command. When the **telnet** command is in character-at-a-time mode and **localchars** has a value of **true**, typing the erase character sends the TELNET EC sequence to the remote host. The initial value for the erase character is the local terminal ERASE character.

escape
Specifies the **telnet escape** character, which puts the **telnet** command into command mode when connected to a remote host. This character can also be specified in octal in the **TNESC** environment variable.

flushoutput
Defines the flush character for the **telnet** command. When **localchars** has a value of **true**, typing the flushoutput character sends the TELNET AO sequence to the remote host. The initial value for the flush character is Ctrl-O. If the remote host is running AIX, the **flushoutput** variable, unlike the other special characters defined by the **set** subcommand, only works in **localchars** mode since it has no **termio** equivalent.

interrupt
Defines the interrupt character for the **telnet** command. When **localchars** has a value of **true**, typing the interrupt character sends the TELNET IP sequence to the remote host. The initial value for the interrupt character is the local terminal interrupt (INTR) character.

kill Defines the kill character for the **telnet** command. When the **telnet** command is in character-at-a-time mode and **localchars** has a value of **true**, typing the kill character sends the TELNET EL sequence to the remote host. The initial value for the kill character is the local terminal KILL character.

quit Defines the quit character for the **telnet** command. When **localchars** has a value of **true**, typing the quit character sends the TELNET BRK sequence to the remote host. The initial value for the quit character is the local terminal QUIT character.

sak Defines the Secure Attention Key (SAK) for the **telnet** command. When the sak character is entered, the remote system is asked to create a trusted shell. If the remote host does not support the SAK, this sequence has no effect.

status Shows the status of the **telnet** command, including the current mode and the currently connected remote host.

toggle *Arguments* Toggles one or more arguments that control how the **telnet** command responds to events. Possible values are **true** and **false**. Multiple arguments are separated by spaces. The **display** subcommand can be used to query the current setting of each argument. The following arguments can be used:

? Displays valid arguments to **toggle**.

autoflush
If **autoflush** and **localchars** both have a value of **true** and the AO, INTR, and QUIT characters are recognized and transformed into TELNET sequences, the **telnet** command does not display any data on the user's terminal until the remote system acknowledges (with a TELNET **timing mark** option) that it has processed those TELNET sequences. The initial value of **autoflush** is **true** if the terminal has not done an **stty noflsh**, and **false** if it has.

autosynch
If **autosynch** and **localchars** are both **true**, then typing the INTR or QUIT character sends that character's TELNET sequence, followed by the TELNET SYNC sequence. This procedure causes the remote host to discard all previously typed input until both of the TELNET sequences have been read and acted upon. The initial value of this toggle is **false**.

crmod Toggles carriage return mode. When set to **true**, most carriage return characters received from the remote host are mapped into a carriage return followed by a line feed. This mode does not affect the characters typed by the user, only those received from the remote host. This mode is useful when the remote host sends only a carriage return and not a line feed. The initial value of this toggle is **false**.

debug Toggles debugging at the socket level. The initial value of this toggle is **false**.

localchars
Determines the handling of TELNET special characters. When this value is **true**, the ERASE, FLUSH, INTERRUPT, KILL, and QUIT characters are recognized locally and transformed into the appropriate TELNET control sequences (EC, AO, IP, BRK, and EL, respectively). When this value is **false**, these special characters are sent to the remote host as literal characters. The initial value of **localchars** is **true** in line-by-line mode and **false** in character-at-a-time mode.

netdata
Toggles the display of all network data (in hexadecimal format). The data is written to standard output unless a *TraceFile* value is specified with the **-n** flag on the **telnet** command line. The initial value of this toggle is **false**.

options
Toggles the display of internal TELNET Protocol processing options, such as terminal negotiation and local or remote echo of characters. The initial value of this toggle is **false**, indicating that the current options should not be displayed.

lineterm
Toggles the default end-of-line terminator to CR-LF (ASCII carriage-return line-feed). A telnet client running on an ASCII host should have the user configurable option to send either the CR-NUL or CR-LF terminator when the user presses the end-of-line key. The initial value of this toggle is **false**.

z Suspends the TELNET process. To return to the TELNET process, use the **fg** built-in command of the **csh** or **ksh** command.

> **Note:** The **z** subcommand has the same effect as a Ctrl-Z key sequence for any other process. It suspends Telnet execution and returns you to your original login shell.

Authentication

If the system is configured for Kerberos 5 authentication, the telnet client will attempt authentication negotiation. The authentication negotiation used by telnet and the definitions of the options and suboptions for this are defined in rfc 1416.

If the client and server agree on an authentication type, they will exchange authentication information including the account the client wants to access. This will be the local user unless the **-l** flag is set.

If they cannot agree on the authentication information or if it fails, the telnet connection will continue with the standard connection (provided Standard AIX is configured).

The remote host allows access only if all of the following conditions are satisfied:

- The local user has current DCE credentials.
- The remote system accepts the DCE credentials as sufficient for access to the remote account. See the **kvalid_user** function for additional information.

Examples

In the following examples, if you enter the **tn** command instead of the **telnet** command, the command mode prompt is displayed as `tn>`.

1. To log in to the remote host `host1` and perform terminal negotiation, enter:

   ```
   telnet host1
   ```

2. To log in to `host1` as a **vt100** terminal (no terminal type negotiation), choose one of the following methods:

   a. Use the following commands to set the **EMULATE** environment variable for this login session, then enter the **telnet** command:

   ```
    EMULATE=vt100; export EMULATE
   telnet host1
   ```

   b. Use the **-e** flag to set the terminal type for this **telnet** session only:

   ```
   telnet  -e vt100 host1
   ```

3. To log in to a remote host and then check the status of the **telnet** program, enter:

   ```
   telnet host3
   ```

   When the login prompt appears, enter your login ID and password. Press the Ctrl-T key sequence to receive the `telnet>` prompt. Enter the following at the `telnet>` prompt:

   ```
    status
   ```

   Information similar to the following is displayed on your screen:

   ```
   Connected to host3.
   Operating in character-at-a-time mode.
   Escape character is '^]'.
   ```

   Upon completion of the **status** subcommand, press the Enter key to return to the remote prompt.

   Once you have completed your login, you can issue commands. To log out of the system and close the connection, press the Ctrl-D key sequence, or exit.

4. To log in to a remote host using the **tn3270** command, enter:

   ```
   tn3270 hostname
   ```

   The host login screen should be displayed. You can now enter your login ID and password. Once you have completed your login, you can issue commands. To log out of the system and close the connection, press Ctrl-D or exit.

Files

/etc/3270.keys	Defines base 3270-keyboard mapping for use with limited function terminals.

Related Information

The **env** command, **ftp** command, **login** command, **rcp** command, **rexec** command, **rlogin** command, **rsh** command.

The **telnetd** daemon.

The **kvalid_user** function.

The **map3270** file format, **.3270keys** file format.

Communications and networks in *Networks and communication management*.

Conversing with a remote user in *Networks and communication management*.

Authentication and the secure rcmds in *Networks and communication management*.

---

telnetd Daemon

Purpose

Provides the server function for the TELNET protocol.

Syntax

/usr/sbin/telnetd [**-a**] [**-n**] [**-s**]

Description

> **Note:** The **telnetd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/telnetd** daemon is a server that supports the Defense Advanced Research Product Agency (DARPA) standard Telnet Protocol (TELNET). Changes to the **telnetd** daemon should be made using the System Management Interface Tool (SMIT).

Changes to the **telnetd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or /**etc**/**services** file. Typing `telnetd` at the command line is not recommended. The **telnetd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file. By default, the **-a** flag is also turned on.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

When a **telnet** session is started, the **telnetd** daemon sends TELNET options to the client (remote) host to indicate an ability to perform options.

Terminal Negotiation

The **telnetd** daemon requests the terminal type from the client host. On receipt, the **telnetd** daemon checks whether the indicated type is supported on the local system. If not, the daemon requests a terminal type again.

This terminal type negotiation continues until the remote client sends an acceptable terminal type or until the client sends the same type twice in a row, indicating that it has no other types available. When necessary, the **telnetd** daemon refers to the **/etc/telnet.conf** file to translate a client's terminal-type strings into **terminfo** file entries.

> **Note:** Because the **telnetd** daemon allows the sending and receiving of 8-bit ASCII, NLS is supported.

If the remote client sends the TELNET **SAK** command, the **telnetd** daemon passes the local SAK characters through the PTY to invoke the trusted shell.

The **telnetd** daemon supports the following TELNET options:

- Binary
- Echo/no echo
- Support SAK
- Suppress go ahead
- Timing mark
- Negotiate About Window Size (NAWS)
- Authentication

The **telnetd** daemon also recognizes the following options for the remote client:

- Binary
- Suppress go ahead
- Echo/no echo
- Terminal type

The **telnetd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Typing `telnetd` at the command line is not recommended.

Authentication Negotiation

If the system has Kerberos 5 authentication configured, **telnetd** will accept authentication option negotiation. If both agree on Kerberos 5 authentication, the client will pass over the DCE principal and **telnetd** will use the **kvalid_user** routine to determine if the DCE principal should have access to the account. If it passes, no password will be requested.

Manipulating the telnetd Daemon with the System Resource Controller

The **telnetd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **telnetd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled by default in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status or a subsystem, group or subsystems, or a subserver.

Flags

-a	Causes the PTY and socket to be linked directly in the kernel so that the data handling remains in the kernel to improve the performance.
-n	Disables transport-level keep-alive messages. Messages are enabled by default.
-s	Turns on socket-level debugging.

Security

The **telnetd** daemon is a PAM-enabled application with a service name of *telnet*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **telnet** service in **/etc/pam.conf**. The **telnetd** daemon requires **/etc/pam.conf** entries for the **auth**, **account**, **password**, and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **telnet** service:

```
#
# AIX telnet configuration
#
telnet auth      required     /usr/lib/security/pam_aix

telnet account   required     /usr/lib/security/pam_aix

telnet password  required     /usr/lib/security/pam_aix

telnet session   required     /usr/lib/security/pam_aix
```

Examples

Note: The arguments for the **telnetd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **telnetd** daemon, type the following:

   ```
   startsrc  -t telnet
   ```

   This command starts the **telnetd** subserver.
2. To stop the **telnetd** daemon normally, type the following:

   ```
   stopsrc  -t telnet
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **telnetd** daemon and all **telnetd** connections, type the following:

   ```
   stopsrc -t  -f telnet
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **telnetd** daemon, type the following:

   ```
   lssrc -t telnet
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

File

terminfo Describes terminal by capability.

Related Information

The **ftp** command, **kill** command, **lssrc** command, **rcp** command, **refresh** command, **rlogin** command, **rsh** command, **startsrc** command, **stopsrc** command, **telnet** command.

The **kill** command, **lssrc** command, **refresh** command, **startsrc** command, **stopsrc** command, **telnet** command.

The **/etc/inetd.conf** file format, **/etc/telnet.conf** file format.

The **pty** special file.

The **kvalid_user** subroutine.

Transmission control protocol in *Networks and communication management*.

TCP/IP daemons in *Networks and communication management*.

Authentication and the secure rcmds in *Networks and communication management*.

---

termdef Command

Purpose

Queries terminal characteristics.

Syntax

termdef [**-c** | **-l** | **-t**]

Description

The **termdef** command identifies the current display type, the active lines setting, or the current columns setting. This simplifies resetting the lines and columns when you switch fonts as well as resetting the **TERM** environment variable when you switch displays. The terminfo database defines the default number of lines and columns for each display, but the lines and columns can change depending upon which font is currently active. Also, the **TERM** environment variable does not automatically reflect the currently active display.

The flags for the **termdef** command are mutually exclusive. If you use more than one flag with the command, the **termdef** command recognizes and returns the current value for the first flag only. Any other flags are ignored. For example, the **termdef -lc** command returns only the active lines setting for the current display.

Flags

-c Returns the current column value.
-l Returns the current line value.
-t Returns the name of the current display (the default action).

Example

To determine the current value of the **TERM** environment variable, enter:

```
termdef -c
```

File

/usr/bin/termdef Contains the **termdef** command.

test Command

Purpose

Evaluates conditional expressions.

Syntax

test *Expression*

OR

[*Expression* **]**

Description

The **test** command evaluates the *Expression* parameter, and if the expression value is True, returns a zero (True) exit value. Otherwise, the **test** command returns a nonzero (False) exit value. The **test** command also returns a nonzero exit value if there are no parameters.

Notes:

1. In the second form of the command, the **[ ]** (brackets) must be surrounded by blank spaces.
2. You must test explicitly for file names in the C shell. File-name substitution (globbing) causes the shell script to exit.

Functions and operators are treated as separate parameters by the **test** command. The *Expression* parameter refers to a statement that is checked for a true or false condition. The following functions are used to construct this parameter:

-b *FileName*	Returns a True exit value if the specified *FileName* exists and is a block special file.
-c *FileName*	Returns a True exit value if the specified *FileName* exists and is a character special file.
-d *FileName*	Returns a True exit value if the specified *FileName* exists and is a directory.
-e *FileName*	Returns a True exit value if the specified *FileName* exists.
-f *FileName*	Returns a True exit value if the specified *FileName* exists and is a regular file.
-g *FileName*	Returns a True exit value if the specified *FileName* exists and its Set Group ID bit is set.
-h *FileName*	Returns a True exit value if the specified *FileName* exists and is a symbolic link.
-k *FileName*	Returns a True exit value if the specified *FileName* exists and its sticky bit is set.
-L *FileName*	Returns a True exit value if the specified *FileName* exists and is a symbolic link.
-n *String1*	Returns a True exit value if the length of the *String1* variable is nonzero.
-p *FileName*	Returns a True exit value if the specified *FileName* exists and is a named pipe (FIFO).
-r *FileName*	Returns a True exit value if the specified *FileName* exists and is readable by the current process.
-s *FileName*	Returns a True exit value if the specified *FileName* exists and has a size greater than 0.
-t *FileDescriptor*	Returns a True exit value if the file with a file descriptor number of *FileDescriptor* is open and associated with a terminal.
-u *FileName*	Returns a True exit value if the specified *FileName* exists and its Set User ID bit is set.

-w *FileName*	Returns a True exit value if the specified *FileName* exists and the write flag is on. However, the *FileName*will not be writable on a read-only file system even if **test** indicates true.
-x *FileName*	Returns a True exit value if the specified *FileName* exists and the execute flag is on. If the specified file exists and is a directory, the True exit value indicates that the current process has permission to search in the directory.
-z *String1*	Returns a True exit value if the length of the *String1* variable is 0 (zero).
*String1***=** *String2*	Returns a True exit value if the *String1* and *String2* variables are identical.
*String1***!=***String2*	Returns a True exit value if the *String1* and *String2* variables are not identical.
String1	Returns a True exit value if the *String1* variable is not a null string.
Integer1 **-eq** *Integer2*	Returns a True exit value if the *Integer1* and *Integer2* variables are algebraically equal. Any of the comparisons **-ne**, **-gt**, **-ge**, **-lt**, and **-le** can be used in place of **-eq**.
file1 **-nt** *file2*	True if *file1* is newer than *file2*.
file1 **-ot** *file2*	True if *file1* is older than *file2*.
file1 **-ef** *file2*	True if *file1* is another name for *file2*.

These functions can be combined with the following operators:

!	Unary negation operator
-a	Binary AND operator
-o	Binary OR operator (that is, the **-a** operator has higher precedence than the **-o** operator)
\(*Expression***\)**	Parentheses for grouping

Exit Status

This command returns the following exit values:

0	The *Expression* parameter is true.
1	The *Expression* parameter is false or missing.
>1	An error occurred.

Examples

1. To test whether a file exists and is not empty, type:

   ```
   if test ! -s "$1"
   then
      echo $1 does not exist or is empty.
   fi
   ```

   If the file specified by the first positional parameter to the shell procedure, `$1`, does not exist, the **test** command displays an error message. If `$1` exists and has a size greater than 0, the **test** command displays nothing.

   Note: There must be a space between the **-s** function and the file name.

   The quotation marks around `$1` ensure that the test works properly even if the value of `$1` is a null string. If the quotation marks are omitted and `$1` is the empty string, the **test** command displays the error message `test: argument expected`.

2. To do a complex comparison, type:

   ```
   if [ $# -lt 2  -o  ! -e "$1" ]
   then
      exit
   fi
   ```

If the shell procedure is given fewer than two positional parameters or the file specified by $1 does not exist, then the shell procedure exits. The special shell variable $# represents the number of positional parameters entered on the command line that starts this shell procedure.

The Shells in *Operating system and device management* describes shells in general, defines terms that are helpful in understanding shells, and describes the more useful shell functions.

File

/usr/bin/test	Contains the **test** command.

Related Information

The **bsh** command, **csh** command, **find** command, **ksh** command, **sh** command.

Shells in *Operating system and device management*.

---

tftp or utftp Command

Purpose

Transfers files between hosts using the Trivial File Transfer Protocol (TFTP).

Syntax

{**tftp** | **utftp**} { **-g** | **-o** | **-p** | **-r** | **-w** } *LocalName HostPort RemoteName* [**netascii** | **image**] [**blksize** *#*] [**timeout** *#*] [**tsize**]

Interactive Form Syntax

Command Line Form Syntax

Description

The **/usr/bin/tftp** and **utftp** commands transfer files between hosts using the Trivial File Transfer Protocol (TFTP). Since TFTP is a minimal file transfer protocol, the **tftp** and **utftp** commands do not provide all of the features of the **ftp** command. For example, the **tftp** and **utftp** commands do not provide the ability to list remote files or change directories at the remote host, and only limited file access privileges are given to the remote TFTP server. The **utftp** command is a form of the **tftp** command for use in a pipe.

The remote host must have a **tftpd** daemon started by its **inetd** daemon and have an account defined that limits the access of the **tftpd** daemon. Use the procedure defined by the **tftpd** command to setup the TFTP environment and the nobody account.

> **Note:** The **tftp** and **utftp** commands should not be available when your host is operating in secure mode.

The **tftp** command ignores duplicate acknowledgments for any block sent and sends an error packet and exit if a block with an inappropriate (future) block number arrives. It also ignores duplicate data blocks if they have already been received and sends an error packet and exits.

RFC2349 Option Negotiation

As of AIX 5.3, the **tftp** client is capable of negotiating the following TFTP options with the server: block size (**blksize**), transfer size (**tsize**), and timeout (**timeout**). Larger transfer block size can improve transfer

performance, **tsize** reports the file size before the transfer to check for available space, and **timeout** negotiates the retransmit timeout. The TFTP server must support RFC2349 for option negotiation to take place.

Access Control

The **/etc/tftpaccess.ctl** file is searched for lines that start with `allow:` or `deny:`. Other lines are ignored. If the file doesn't exist, access is allowed. The allowed directories and files can be accessed and the denied directories cannot be accessed. For example, the **/usr** directory might be allowed and the **/usr/ucb** directory might be denied. This means that any directory or file in the **/usr** directory, except the **/usr/ucb** directory, can be accessed. The entries in the **/etc/tftpaccess.ctl** file must be absolute path names.

The **/etc/tftpaccess.ctl** file should be write-only by the root user and readable by all `groups` and `others` (that is, owned by `root` with permissions of 644). The user `nobody` must be able to read the **/etc/tftpaccess.ctl** file. Otherwise, the **tftpd** daemon is not able to recognize the existence of the file and allows access to the entire system. For more information, refer to the sample **tftpaccess.ctl** file, which resides in the **/usr/samples/tcpip** directory.

The search algorithm assumes that the local path name used in the **tftp** command is an absolute path name. It searches the **/etc/tftpaccess.ctl** file looking for `allow:/`. It repeatedly searches for allowed path names with each partial path name constructed by adding the next component from the file path name. The longest path name matched is the one allowed. It then does the same with denied names, starting with the longest allowed path name matched.

For example, if the file path name were **/a/b/c** and the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and `deny:/a`, one allowed match would be made (`/a/b`) and no denied match starting with `/a/b` would be made, and access would be allowed.

If the **/etc/tftpaccess.ctl** file contained `allow:/a` and `deny:/a/b`, one allowed match would be made (`/a`) and one denied match starting with `/a` (`/a/b`) would be made, and access would be denied. If the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and also contained `deny:/a/b`, access would be denied because allowed names are searched first.

> **Note:** Further information and example configurations for Xstations, Diskless clients, and restricted entry can be found in the **/usr/samples/tcpip/tftpaccess.ctl** file.

The **tftp** and **utftp** commands have two forms: interactive form and command-line form.

Interactive Form

In the interactive form, the **tftp** and **utftp** commands are issued alone or with a *Host* parameter that specifies the default host to use for file transfers during this session. If you choose, you can also specify with the *Port* parameter which port the **tftp** or **utftp** connection should use, such as the one specified for **mail** in the **/etc/services** file. When you enter the interactive form of either of these commands, the `tftp>` prompt is displayed.

When transferring data to a remote host, the transferred data is placed in the directory specified by the *RemoteName* parameter. The remote name must be a fully specified file name, and the remote file must both exist and have write permission set for others. The **tftp** command attempts to write the data to the specified file. However, if the remote TFTP server does not have the appropriate privileges to write the remote file or if the file does not already exist, the transfer is unsuccessful. This can be overridden using the **tftpd** daemon.

Command-Line Form

The command-line forms of the **tftp** and **utftp** commands are equivalent, except that the **utftp** command does not overwrite a local file. The **tftp** command can overwrite a file, but prompts the user before doing so. Because it is not interactive, the command line form of the **utftp** command can be more useful than

the **tftp** command in a pipe. In the command line form, all of the arguments to either command are specified on the command line, and no prompt is displayed.

Subcommands

The **tftp** and **utftp** subcommands can be entered in either their interactive form or in their command-line form.

Subcommands Used in the Interactive Form

Once the `tftp>` prompt is displayed, the following subcommands can be issued:

? [*Subcommand*]
Displays help information. If a *Subcommand* parameter is specified, only information about that subcommand is displayed.

ascii
Synonym for the **mode ascii** subcommand.

binary
Synonym for the **mode binary** subcommand. This subcommand is used in the interactive mode. The **image** subcommand accomplishes the same thing as the **mode binary** subcommand, but is used on the command line.

blksize *Number of Bytes*
Enables the **blksize** option negotiation with the server. If successfully negotiated, this can substantially improve transfer rates. The transfer block size must be at least 8 octets and can be as high as 65464 octets. The default is 512 octets.

connect *Host* [*Porf*]
Sets the remote host, and optionally the port, for file transfers. Since the TFTP protocol does not maintain connections between transfers, the **connect** subcommand does not create a connection to the specified host, but stores it for transfer operations. Because the remote host can be specified as part of the **get** or **put** subcommand, which overrides any host previously specified, the **connect** subcommand is not required.

get *RemoteFile* [*LocalFile*]

get *RemoteFile RemoteFile RemoteFile* [*RemoteFile* . . .]
Gets a file or set of files from the remote host to the local host. Each of the *RemoteFile* parameters can be specified in one of the following two ways:

- As a file (*File*) that exists on the remote host if a default host has already been specified.
- As a host file (*Host*:*File*), where *Host* is the remote host and *File* is the name of the file to copy to the local system. If this form of the parameter is used, the last host specified becomes the default host for later transfers in this **tftp** session.

mode *Type*
Sets the type (*Type*) of transfer mode to either **ascii** or **binary**. A transfer mode of **ascii** is the default.

put *LocalFile* [*RemoteFile*]

put *LocalFile LocalFile LocalFile* [*LocalFile* . . .] *RemoteDirectory*

Puts a file or set of files from the local host onto the remote host. The *RemoteDirectory* and *RemoteFile* parameters can be specified in one of the following two ways:

- As a file or directory that exists on the remote host if a default host has already been specified.
- With *Host*:*RemoteFile* parameter, where *Host* is the remote host and *RemoteFile* is the name of the file or directory on the remote system. If this form of the parameter is used, the last host specified becomes the default host for later transfers in this **tftp** session.

In either case, the remote file or directory name must be a fully specified path name, even if the local and remote directories have the same name. If a remote directory is specified, the remote host is assumed to be a UNIX machine. The default value of the **put** subcommand is write-replace, but you can add an option in the **tftpd** daemon to allow write-create.

quit Exits the **tftp** session. An End-Of-File key sequence also exits the program.

status Shows the current status of the **tftp** program, including, for example, the current transfer mode (**ascii** or **binary**), connection status, and time-out value.

timeout *Value* Sets the total transmission time out to the number of seconds specified by the *Value* parameter. The *Value* parameter must be 1 second or greater (the default is 5 seconds).

trace Turns packet tracing on or off.

tsize Enables the **tsize** option negotiation with the server. This allows the file size to be known before the transfer starts. If allocation is exceeded, an error is returned and the file transfer does not occur.

verbose Turns verbose mode, which displays additional information during file transfer, on or off.

Subcommands Used in the Command Line Form

In this form, if the *Action* flag is:

-w or **-p** Writes (or puts) local data, specified by the *LocalName* parameter, to the file specified by the *RemoteName* parameter on the remote host specified by the *Host* parameter. If the *LocalName* parameter is a file name, the **tftp** command transfers the specified local file. If the *LocalName* parameter is specified as a - (dash), the **tftp** command transfers data from local standard input to the remote host. When the *LocalName* parameter is standard input, the **tftp** command allows 25 seconds for all input to be entered before it times out.

-r or **-g** or **-o** Reads (or gets) remote data from the file specified by the *RemoteName* parameter at the remote host specified by the *Host* parameter and writes it to the file specified by the *LocalName* parameter. If the *LocalName* parameter is a file name, the **tftp** command writes the data to the specified local file. For the **-r** and **-g** actions, the **tftp** command prompts for verification before overwriting an existing local file. For the **-o** action, the **tftp** command overwrites an existing local file without prompting. If the *LocalName* parameter is specified as a - (dash), the **tftp** command writes the data to local standard output.

> **Note:** Since the **tftp -g** and **tftp -r** commands prompt before overwriting an existing local file, it may be impractical to use the **tftp** command in a pipe. The **utftp** command performs the same **-r** and **-g** actions as the **tftp** command, but simply stops before overwriting a local file. Thus, the **utftp** command may be more appropriate for use in a pipe.

For both of the following modes of file transfer, the *RemoteName* parameter is the name of a file that has write permission set for others. Note that the *RemoteName* parameter must be in double quotes (″ ″) if it contains shell special characters.

The mode of transfer is one of the following:

netascii Transfers the data as 7-bit ASCII characters in 8-bit transfer bytes. This is the default.

image Transfers the data as 8-bit binary data bytes in 8-bit transfer bytes, with no conversion. **image** transfer can be more efficient than **netascii** transfer when transferring between two hosts. It is recommended that **netascii** be used when transferring ASCII files from a workstation to a different type of host.

Examples

The following examples distinguish the differences between the interactive form and the command line form of the **tftp** command:

Using the Interactive Form of the tftp Command

To enter the **tftp** command, check the current status, connect to a remote host, and transfer a file from a remote host to your local host, enter:

```
tftp
```

The `tftp>` prompt is displayed. Enter the **status** subcommand following this prompt:

```
 status
```

A message similar to the following is displayed on your screen:

```
Not connected.
Mode: netascii   Verbose: off   Tracing: off
Max-timeout: 25 seconds
tftp> _
```

After the `tftp>` prompt, enter the **connect** subcommand and the name of the remote system to which you want to connect:

```
tftp>  connect host1
```

The `tftp>` prompt is displayed as an indication that you are connected to `host1`. Following the `tftp>` prompt, enter the **get** subcommand to transfer the file `update` from the remote host to your local host.

```
 get /home/alice/update update
```

The `/home/alice` directory on the remote host must have read permission set for others. The `/home/alice/update` file from `host1` was transferred to the `update` file on your local system. In this example, the user is connected to `host1` and the `update` file is transferred from `host1` to the local host.

Using the Command Line Form of the tftp Command

1. To copy a text file from a remote host and write it to a local file, enter:

   ```
   tftp  -g newsched host1 /home/john/schedule
   $ _
   ```

   In this example, the `/home/john/schedule` file was copied from the remote host `host1` and written to the local file `newsched`.
2. To copy a file from a remote host and redirect the output to standard output of the local host, enter:

```
tftp  -g - host3 /etc/hosts
```

If the copy is successful, information similar to the following is displayed on your screen:

```
192.100.13.3 nameserver
192.100.13.3 host2
192.100.13.5 host1
192.100.13.7 host3
192.100.13.3 timeserver
Received 128 bytes in 0.4 seconds
$ _
```

In this example, the `/etc/hosts` file from remote host `host3` was copied and the output redirected to standard output of the local host.

3. To copy a file from a remote host, pipe it to the **grep** command, and write it to a local file, enter:

```
utftp  -g - host1 /home/john/schedule | grep Jones > jones.todo
$ _
```

In this example, the `/home/john/schedule` file was copied from the remote host `host1`. This file was then piped to the **grep** command and written into the local file `jones.todo`.

4. To copy a file to another system, enter:

```
tftp  -p /home/jeanne/test host2 /tmp/test
```

If the copy is successful, information similar to the following is displayed on your screen:

```
Sent 94146 bytes in 6.7 seconds
```

In this example, the `/home/jeanne/test` file was sent to the `/tmp` directory on the remote host `host2`.

5. To copy a binary file to another system, enter:

```
tftp  -p core host3 /tmp/core image
```

If the copy is successful, information similar to the following is displayed on your screen:

```
Sent 309295 bytes in 15 seconds
```

In this example, the binary file `core` from the current directory was sent to the `/tmp` directory on remote host `host3`.

Files

/etc/tftpaccess.ctl	Allows or denies access to files and directories.

Related Information

The **ftp** command, **grep** command, **rcp** command.

The **ftpd** daemon, **inetd** daemon, **tftpd** daemon, **syslogd** daemon.

The **hosts** file format, **services** file format.

File transfers using the tftp and utftp commands in *Networks and communication management*.

Communications and networks in *Networks and communication management*.

tftpd Daemon

Purpose

Provides the server function for the Trivial File Transfer Protocol.

Syntax

/usr/sbin/tftpd [**-c**] [**-n**] [**-p**] [**-v**] [**-t**] [**-s**] [**-x**] [**-z**] [**-d** *Directory*] [**-r** *Option*]

Description

> **Note:** The **tftpd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

The **/usr/sbin/tftpd** daemon runs the Trivial File Transfer Protocol (TFTP) server. Files sent using TFTP can be found in the directory specified by the full path name given on the **tftp** or **utftp** command line.

> **Note:** The **tftp** command, **utftp** command, and **tftpd** server are not available when the auditing system is in use. For more information, see TCP/IP Security, the Auditing overview, and the **audit** command.

Changes to the **tftpd** daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. The **tftpd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon get its information from the **/etc/inetd.conf** file and the **/etc/services** file.

After changing the **/etc/inetd.conf** or **/etc/services** file, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

The **tftpd** server should have a user ID with the least privileges possible. The **nobody** ID allows the least permissions, and is the default user ID.

The **tftpd** daemon should be controlled using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Entering `tftpd` at the command line is not recommended.

With AIX 4.3.3, the **tftpd** server has become a multithreaded application. Another major change with AIX 5.3 is the server's ability to handle option negotiation (RFC2349). This new capability allows a client to negotiate a file size to be transferred. It also allows for a timeout and a larger block size. Block size (**blksize**) is negotiated for the read requests (RRQ) only. As a result, the boot time performance of diskless nodes using TFTP can improve significantly.

The Transfer Size option (**tsize**) negotiation for both read and write requests allows the file size to be known before the transfer, resulting in an error message if allocation exceeded before the transfer started. The timeout option (**timeout**) allows for the client and the server to negotiate a retransmit timeout (between 1 and 255 seconds). The **tftp** client must also support RFC2349 for the option negotiation to take place.

tftpaccess.ctl File

The **/etc/tftpaccess.ctl** file is searched for lines that start with `allow:` or `deny:`. Other lines are ignored. If the file doesn't exist, access is allowed. The allowed directories and files minus the denied directories and files can be accessed. For example, the **/usr** directory might be allowed and the **/usr/ucb** directory might be denied. This means that any directory or file in the **/usr** directory, except the **/usr/ucb** directory, can be accessed. The entries in the **/etc/tftpaccess.ctl** file must be absolute path names.

The **/etc/tftpaccess.ctl** file should be write-only by the root user and readable by all `groups` and `others` (that is, owned by `root` with permissions of 644). The user `nobody` must be able to read the **/etc/tftpaccess.ctl** file. Otherwise, the **tftpd** daemon is not able to recognize the existence of the file and allows access to the entire system. For more information, refer to the sample **tftpaccess.ctl** file, which resides in the **/usr/samples/tcpip** directory.

The search algorithm assumes that the local path name used in the **tftp** command is an absolute path name. It searches the **/etc/tftpaccess.ctl** file looking for `allow:/`. It repeatedly searches for allowed path names with each partial path name constructed by adding the next component from the file path name. The longest path name matched is the one allowed. It then does the same with denied names, starting with the longest allowed path name matched.

For example, if the file path name were **/a/b/c** and the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and `deny:/a`, one allowed match would be made (`/a/b`) and no denied match starting with `/a/b` would be made, and access would be allowed.

If the **/etc/tftpaccess.ctl** file contained `allow:/a` and `deny:/a/b`, one allowed match would be made (`/a`) and one denied match starting with `/a` (`/a/b`) would be made, and access would be denied. If the **/etc/tftpaccess.ctl** file contained `allow:/a/b` and also contained `deny:/a/b`, access would be denied because allowed names are searched first.

Manipulating the tftpd Daemon with the System Resource Controller

The **tftpd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **tftpd** daemon is a member of the **tcpip** SRC subsystem group. This daemon is enabled when it is uncommented in the **/etc/inetd.conf** file and can be manipulated by the following SRC commands:

startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the status of a subsystem, group of subsystems, or a subserver.

Flags

-c	Specifies the maximum number of concurrent threads per process, excluding the initial thread.
-d *Directory*	Specifies default destination directory. The *Directory* specified will be used as the home directory for storing files only. This default directory will be used only if a full pathname is not specified. The default directory for retrieving files is still **/tftpboot**.
-i	Logs the IP address of the calling machine with error messages.
-n	Allows the remote user to create files on your machine. Remote users are only allowed to read files with read permission for other if this flag is not specified.
-p	Specifies the port number for the incoming request.
-r *Option*	Specifies a **tftp** option negotiation to disable. Multiple **-r** flags can be used. For example, the following line in the **/etc/inetd.conf** file disables option negotiation for **tsize** and **blksize**: `tftp dgram udp6 SRC nobody /usr/sbin/tftpd tftpd -n -r tsize -r blksize`
-s	Turns on socket-level debugging.
-t	Specifies the timeout value for datagrams.
-v	Logs information messages when any file is successfully transferred by the **tftpd** daemon. This logging keeps track of who is remotely transferring files to and from the system with the **tftpd** daemon.
-x	Specifies the maximum of timeouts waiting for a datagram.
-z	Specifies the maximum allowed segment size for transfers.

Examples

Note: The arguments for the **tftpd** daemon can be specified by using SMIT or by editing the **/etc/inetd.conf** file.

1. To start the **tftpd** daemon, enter the following:

   ```
   startsrc  -t tftp
   ```

   This command starts the **tftpd** subserver.
2. To stop the **tftpd** daemon normally, enter the following:

   ```
   stopsrc  -t tftp
   ```

   This command allows all pending connections to start and existing connections to complete but prevents new connections from starting.
3. To force stop the **tftpd** daemon and all **tftpd** connections, enter the following:

   ```
   stopsrc -t  -f tftp
   ```

   This command terminates all pending connections and existing connections immediately.
4. To display a short status report about the **tftpd** daemon, enter the following:

   ```
   lssrc -t tftp
   ```

   This command returns the daemon's name, process ID, and state (active or inactive).

Related Information

The **kill** command, **lssrc** command, **refresh** command, **startsrc** command, **stopsrc** command, **tftp** command.

The **inetd** daemon.

The **/etc/inetd.conf** file format.

Trivial File Transfer Protocol (TFTP) in *Networks and communication management*.

TCP/IP daemons in *Networks and communication management*.

tic Command

Purpose

Translates the terminfo description files from source to compiled format.

Syntax

tic [**-v** [*Number*]] [**-c**] *FileName*

Description

The **tic** command translates the terminfo files from the source format into the compiled format. The **tic** command places the results in the **/usr/share/lib/terminfo** directory. If the **TERMINFO** environment variable is set, the results are placed there instead of in the **/usr/share/lib/terminfo** directory.

The **tic** command compiles all terminfo descriptions in *FileName*. When the **tic** command finds a `use=` entry-*name* field, it searches the current file first, If unable to find the entry -*name*, it obtains the entry from the binary file in **/usr/share/lib/terminfo**. If **TERMINFO** is set, the terminfo directory is searched before **/usr/share/lib/terminfo**.

The total compiled entries cannot exceed 4096 bytes, and the name field cannot exceed 128 bytes.

Flags

-v[*Number*] Writes trace information on the progress of the **tic** command. *Number* is an integer from 1 to 10 inclusive that increases the level of the verbosity. If *Number* is omitted, the default level is 1. The amount of information output increases as *Number* increases.

-c Only checks *FileName* for errors. Errors in *use=entry-name* are not detected.

Files

/usr/share/lib/termi

nfo/?/* Contains the compiled terminal capability database.

Related Information

The **terminfo** file format.

The Curses Overview for Programming in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

time Command

Purpose

Prints the time of the execution of a command.

Syntax

time [**-p**] *Command* [*Argument* ...]

Description

The **time** command prints the elapsed time during the execution of a command, time in the system, and execution time of the **time** command in seconds to standard error.

Note: Sleep time is not charged to either system or user time.

The **time** command is also built into the C shell (**csh**) and Korn shell (**ksh**) with a different format. To run the **time** command while in the **csh** and **ksh** shells, enter:

```
/usr/bin/time
```

Flags

-p Writes the timing output to standard error. Seconds are expressed as a floating-point number with at least one digit following the radix character.

The standard format for this flag is as follows:

```
"real %f\nuser %f\nsys %f\n", <real seconds>, <user seconds>, <system seconds>
```

Exit Status

If you use the *Command* parameter, the exit status of the **time** command is the exit status of the specified command. Otherwise, the **time** command exits with one of the following values:

1-125 Indicates an error occurred in the **time** command.
126 Indicates the command specified by the *Command* parameter was found but could not be invoked.
127 Indicates the command specified by the *Command* parameter could not be found.

Examples

1. To measure the time required to run a program, enter:

   ```
   /usr/bin/time -p a.out
   ```

   This command runs the program **a.out** and writes the amount of real, user, and system time to standard error, in the format specified by the **-p** flag; for example:

   ```
   real      10.5
   user      0.3
   sys       3.6
   ```

2. To save a record of the **time** command information in a file, enter:

   ```
   /usr/bin/time a.out 2> a.time
   ```

Files

/usr/bin/time Specifies the path of the **time** command.

Related Information

The **timex** command.

Setting up an accounting subsystem in *Operating system and device management*.

Accounting Commands in *Operating system and device management*.

System accounting in *Operating system and device management*.

Using the time Command to Measure CPU Use in *Performance management*.

---

timed Daemon

Purpose

Invokes the time server daemon.

Syntax

/usr/sbin/timed [**-c**] [**-M**] [**-t**] [[**-n** *Network*] ... | [**-i** *Network*] ...]

> **Note:** Use the **rc.tcpip** file to start the daemon with each initial program load. You can specify the **timed** daemon at the command line. You can also use SRC commands to control the **timed** daemon from the command line.

Description

The **timed** daemon synchronizes one machine's clock with those of other machines on the local area network that are also running the **timed** daemon. The **timed** daemon slows the clocks of some machines and speeds up the clocks on other machines to create an average network time.

When the **timed** daemon is started without the **-M** flag, the machine locates the nearest master time server and asks for the network time. Then the machine uses the **date** command to set the machine's clock to the network time. The machine accepts synchronization messages periodically sent by the master time server and calls the **adjtime** subroutine to perform the needed corrections on the machine's clock.

When the **timed** daemon is started with the **-M** flag, the machine polls each of its local area networks to determine which networks have master time servers. The machine becomes a master time server on the networks that do not have a master time server. The machine becomes a submaster time server on the networks that already have a master time server. The **timed** daemon creates the **/var/adm/timed.masterlog** file when the **timed** daemon is started with the **-M** flag. The **/var/adm/timed.masterlog** file contains a log of the deltas between the local machine's clock and the clocks of the other machines on the networks for which the local machine is the master time server. The **/var/adm/timed.masterlog** file is updated approximately every 4 minutes and is never cleared. You may need to clear this file to conserve disk space. If the machine is only a submaster time server on its networks, the **/var/adm/timed.masterlog** file remains empty. To clear the **/var/adm/timed.masterlog** file, enter:

```
cat /dev/null > /var/adm/timed.masterlog
```

If the master time server ceases to function on a network, a new master time server is elected from the submaster time servers on that network. The **timedc** command enables you to select which submaster time server becomes the master time server.

The **timed** daemon can be controlled using the System Resource Controller (SRC), the System Management Interface Tool (SMIT), or the command line. The **timed** daemon is not started by default. Use the **rc.tcpip** file to start the **timed** daemon with each initial program load.

Manipulating the timed Daemon with the System Resource Controller

The **timed** daemon is a subsystem controlled by the SRC. The **timed** daemon is a member of the SRC **tcpip** system group. Use the following SRC commands to manipulate the **timed** daemon:

startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
lssrc	Gets the short status of a subsystem, group of subsystems, or a subserver. The long status option usually found in **lssrc** is not supported for the timed daemon.

Flags

-c	Specifies that the master-timed daemon should ignore the time values it gets from the other slave-timed daemons when for calculating the average network time. This flag changes the network time to be the same as the system clock on the master-timed daemon.
-i *Network*	Specifies a network to be excluded from clock synchronization. The *Network* variable can be either a network address or a network name. If a network name is specified for the *Network* variable, the network name must be defined in the **/etc/networks** file. Specify one network address or network name with each **-i** flag. Do not use this flag with the **-n** flag.
-M	Specifies the machine is a master or submaster time server on its local area networks. If a master time server is not currently available on a network, the machine becomes the master time server for that network. If a master time server already exists on a network, the machine becomes a submaster time server on that network. However, the machine can become the master time server if the current master time server becomes inoperative. The **timed** daemon creates the **/var/adm/timed.masterlog** file when the **timed** daemon is started with the **-M** flag.

-n *Network* Specifies a network to include in clock synchronization. The *Network* variable can be either a network address or a network name. If a network name is specified for the *Network* variable, the network name must be defined in the **/etc/networks** file. Specify one network address or network name with each **-n** flag. Do not use this flag with the **-i** flag.

-t Allows the **timed** daemon to trace the messages it receives and store them in the **/var/adm/timed.log** file. You can also use the **timedc** command to activate tracing.

Examples

1. To start the **timed** daemon with SRC control, enter:

   ```
   startsrc -s timed
   ```

   This command starts the daemon. You can use this command in the **rc.tcpip** file or on the command line. The **-s** flag specifies that the subsystem that follows is to be started.
2. To stop the **timed** daemon normally with SRC control, enter:

   ```
   stopsrc -s timed
   ```

   This command stops the daemon. The **-s** flag specifies that the subsystem that follows is to be stopped.
3. To get a short status report from the **timed** daemon, enter:

   ```
   lssrc -s timed
   ```

   This command returns the name of the daemon, the process ID of the daemon, and the state of the daemon (active or inactive).
4. To start the **timed** daemon with SRC control as the master or submaster time server and to exclude networks `net1` and `net2` from clock synchronization, enter:

   ```
   startsrc -s timed -a "-M -i net1 -i net2"
   ```

   This command starts the daemon. The machine becomes the master or submaster time server for its networks. Networks `net1` and `net2` are excluded from clock synchronization. The **-s** flag specifies that the subsystem that follows is to be started. The **-a** flag specifies that the **timed** daemon should be started with the flags that follow. The flags must be enclosed in quotes.
5. To start the **timed** daemon, activate tracing, and include `net1` and `net2` in clock synchronization, enter:

   ```
   timed -t -n net1 -n net2
   ```

   This command starts the daemon. Tracing is activated and both `net1` and `net2` are included in clock synchronization.

Files

/var/adm/timed.log Contains the messages traced for the **timed** daemon. This file is created when the **timed** daemon is started with the **-t** flag or when tracing is enabled with the **timedc** command.

/etc/rc.tcpip Contains the SRC commands to be executed at system startup.

/var/adm/timed.masterlog Contains a log of the deltas between the master time server clock and the clocks of the other machines on the networks. This file is created when the **timed** daemon is started with the **-M** flag. However, this file only contains information for those networks on which the machine is the master time server.

Related Information

The **date** command, **timedc** command.

The **adjtime** subroutine, **gettimeofday** subroutine.

The **networks** file format.

TCP/IP daemons in *Networks and communication management*.

---

timedc Command

Purpose

Returns information about the **timed** daemon.

Syntax

timedc [*Subcommand* [*Parameter* ...]]

Description

The **timedc** command controls the operation of the **timed** daemon. The **timedc** command does the following:

- Measures the difference between clocks on various machines on a network.
- Finds the location of the master time server.
- Enables or disables tracing of messages received by the **timed** daemon.
- Debugs.

Without any variables, the **timedc** command assumes an interactive mode and prompts for subcommands from standard input. If variables are supplied, the **timedc** command interprets the first variable as a subcommand and the remaining variables as parameters to the subcommand. You can redirect standard input so the **timedc** command reads subcommands from a file.

Variables

The **timedc** command recognizes the following subcommands:

? [*Parameter* ...]	Displays a short description of each variable specified in the parameter list. The **?** subcommand only works in interactive mode. If you give no variables, the **?** subcommand shows a list of subcommands recognized by the **timedc** command.
clockdiff *Host* ...	Computes the differences between the clock of the host machine and the clocks of the machines given as variables.
election *Host* ...	Requests that the **timed** daemon on the specified host (s) reset its election timers and ensure that a **timed** master server is available. Up to 4 hosts can be specified. If a master **timed** server is no longer available, then the **timed** daemon on the specified host (s) will request to become the new **timed** master server. The specified host(s) must be running the **timed** daemon in submaster mode with the **-M** flag.
help [*Parameter* ...]	Displays a short description of each subcommand specified in the parameter list. If you give no variables, the **help** subcommand shows a list of subcommands recognized by the **timedc** command.
msite	Finds the location of the master site.
quit	Exits the **timedc** command.

trace { **on** | **off** }	Enables or disables tracing of incoming messages to the **timed** daemon. The messages are held in the **/var/adm/timed.log** file.

You can use other commands for testing and debugging the **timed** daemon. Use the **help** command to find these commands.

These error messages may occur with the **timedc** command:

`Ambiguous command`	Abbreviation matches more than one command.
`Invalid command`	No match found.
`Privileged command`	Command can be executed only by the root user.

Examples

1. To display the time difference between the local host `sahara` and the remote host `sandy`, type:

   ```
   timedc clockdiff sandy
   ```

   The output would be:

   ```
   time on sandy.austin.century.com is 37904247 ms ahead of time on
   sahara.austin.century.com
   ```

2. To display the client location of the **timed** daemon, type:

   ```
   timedc msite
   ```

   The output would be:

   ```
   client timed daemon runs on bupu.austin.century.com
   ```

Related Information

The **date** command.

The **timed** daemon.

The **adjtime** subroutine.

TCP/IP daemons in *Networks and communication management*.

---

timex Command

Purpose

Reports, in seconds, the elapsed time, user time, and system execution time for a command.

Syntax

timex [**-o**] [**-p**] [**-s**] *Command*

Description

The **timex** command reports, in seconds, the elapsed time, user time, and system execution time for a command. With specified flags, the **timex** command lists or summarizes process accounting data for a command and all of its children. *Command* is the name of any executable file on the system. It also reports total system activity during the execution interval. Output is written to standard error. The system uses the **/var/adm/pacct** file to select process records associated with the command and includes background processes with the same user ID, workstation ID, and execution time window.

Flags

-o Reports the total number of blocks read or written and total characters transferred by a command and all its children.

-p Lists process accounting records for a command and all its children. The number of blocks read or written and the number of characters transferred are reported. The **-p** flag takes the **f**, **h**, **k**, **m**, **r**, and **t** arguments defined in the **acctcom** command to modify other data items.

- **-f** Print the fork/ exec flag and system exit status columns in the output.
- **-h** Instead of mean memory size, shows the fraction of total available CPU time consumed by the process (hogfactor).
- **-k** Instead of memory size, shows total kcore minutes (memory measurement in kilobyte segments used per minute of run time).
- **-m** Shows mean main-memory size. This is the default. The **-h** flag or **-k** flag turn off the **-m** flag.
- **-r** Shows CPU factor.
- **-t** Shows separate system and user CPU times.

-s Reports total system activity during the execution of the command. All the data items listed in the **sar** command are reported.

Note: Accounting must be turned on to use the **-o** or **-p** flags.

Examples

1. To report the total number of blocks read and total characters transferred by the **ls** command, enter:
   ```
   timex -o ls
   ```
2. To list the process accounting records for the **ps** command, enter:
   ```
   timex -p ps -fe
   ```
3. To report total system activity for the execution of the **ls** command, enter:
   ```
   timex -s ls
   ```

Files

/var/adm/pacct Used to select record associated with the command.

Related Information

The **acctcom** command, **sar** command, **time** command.

Setting up an accounting subsystem in *Operating system and device management* describes the steps you must take to establish an accounting system.

See the Accounting commands in *Operating system and device management* for a list of accounting commands that can be run automatically or entered from the keyboard.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in *Operating system and device management*.

Monitoring and tuning commands and subroutines in *Performance management*.

tip Command

Purpose

Connects to a remote system.

Syntax

tip [**-v**] [**-** *BaudRate*] { *SystemName* | *PhoneNumber* }

Description

The **tip** command connects to a remote system and allows you to work on the remote system as if logged in directly.

Either the *SystemName* parameter or the *PhoneNumber* parameter is required. The *SystemName* parameter specifies the name of a remote system to be contacted. The remote system must be defined in the **/etc/remote** file, or in the file specified by the **REMOTE** environment variable. The *PhoneNumber* parameter specifies the number to dial over a modem connection.

When the **tip** command is invoked with the *SystemName* parameter, it searches the **remote** file for an entry beginning with that system name. When the command is invoked with the *PhoneNumber* parameter, it searches the **remote** file for an entry of the form **tip***BaudRate*, where *BaudRate* is the baud rate for the connection. If the **-***BaudRate* flag is not used, the **tip** command looks for a `tip1200` entry, because 1200 is the default baud rate.

The actions of the **tip** command can be controlled using flags, escape signals and variables. The **tip** command reads the **/etc/remote** file to find out how to contact a remote system and discover the escape-send sequence to use when communicating with that system. In addition, the command may check the **/etc/phones** file to find out a phone number for the remote system.

A **tip** user can create an individual remote file in the format of the **/usr/lib/remote-file** file, and then specify the file to use with the **REMOTE** environment variable. A user can also create an individual phones file in the format of the **/usr/lib/phones-file** file, and then specify the file to use with the **PHONES** environment variable. The **tip** command does not read the **/usr/lib/remote-file** file or **/usr/lib/phones-file** file by default, however. The default files that the **tip** command uses are the **/etc/remote** file and **/etc/phones** file.

A **tip** user can create a **$HOME/.tiprc** file to specify initial settings for the **tip** variables. In addition, settings made in the remote file, the phones file, and the **.tiprc** file can be overridden by using escape signals while **tip** is running. Escape signals can also be used, for instance, to start and stop file transfers or interrupt a connection to remote system.

The **tip** command uses lock files in the **/etc/locks** directory to lock devices against multiple access and to prevent multiple users from logging in on the same system.

When the **tip** command prompts for a response, edit the line as you type using the standard keys. Entering ~. (tilde, period) in response to a prompt, or pressing the Interrupt key, will abort the **tip** dialog and return you to the remote system.

You can use the **tip** command to transfer files to and from the remote system. You can use **tip** command escape signals to start and stop the file transfers. Several **tip** command variables work together to control file transfers.

File transfers usually use tandem mode to control the flow of data. If the remote system does not support tandem mode, set the *echocheck* variable to `on` to cause the **tip** command to synchronize with the remote

system after transmitting each character. When transferring files with the ~< and ~> escape signals, use the **eofread** and *eofwrite* variables to specify the end of a file when writing, and recognize the end of a file when reading.

If the *verbose* variable is set `on`, the **tip** command performs the following:

- Writes a running count of the number of lines transferred during a file transfer.
- Writes messages indicating its actions as it dials a phone number.

You can use scripting to record the conversations you have with the **tip** command. Use the *script* variable to start scripting.

Notes:

1. Only a user with root user authority can change the *dialtimeout* variable.
2. Although any user can specify a host at the command line, only the root user can change the *host* variable setting after the **tip** command has been started. However, this does not change the system to which the **tip** command is currently connected.

Flags

-v	Displays the settings of variables as they are read from the **.tiprc** file.
-*BaudRate*	Overrides the default baud rate, which is 1200 baud.

Escape Signals

Using escape signals, you can instruct the **tip** command to terminate, log off from the remote system, and transfer files. The escape character at the beginning of a line indicates an escape signal. The default escape character is a ~ (tilde). The character can be changed using the *escape* variable. All other typed characters are transmitted directly to the remote system. The **tip** command recognizes the following escape signals:

~^D~	Terminates the connection and exits. You may still be logged in on the remote system; if so, you can issue another **tip** command to reconnect to that remote system.
~c [*Directory*]	Changes, on the local system, to the directory specified by the *Directory* variable. If you do not include the *Directory* variable, the **tip** command changes to your home directory.
~!	Escapes to a shell on the local system. When you exit from the shell, you return to the **tip** command.
~>	Copies a file from the local system to the remote system. The **tip** command prompts you for the name of the local file.
~<	Copies a file from the remote system to the local system. The **tip** command prompts you for the name of the remote file.

A **tip** file download will only download the file until one of the EOF characters listed in the **eofread** command cariable is encountered. If one of these characters is not encountered, then the file copy will not succeed.

When downloading a file with the ~< signal, the user will be prompted for a local file name. The user may respond with any valid writeable file name. When prompted for the remote command, the user should append the EOF character to the end of the file being read.

This signal can be used as shown in the following example:

```
List command for remote system? echo "\04" | cat /etc/passwd
```

This example assumes that the character 0x4 is present in the **tip** *eofread* variable. The best way of ensuring that this character exists in the variable is to assign it in the usr's **.tiprc** file, which should reside in the user's home directory.

To accomplish this, the following command can be issued:

```
echo"eofread=\04" >> ~/.tiprc
```

~p *Source* [*Dest*]	Sends (puts) the *Source* file to a remote UNIX host system, using the **cat** command to copy the *Source* file to the *Dest* file. If the *Dest* file name is not specified, the **cat** command uses the name of the *Source* file. If the *Dest* file exists on the remote host, it will be replaced by the *Source* file. This signal is a UNIX-specific version of the ~> signal.
~t *Source* [*Dest*]	Transfers (takes) the *Source* file from a remote UNIX host system to the local system, using the **cat** command to copy the *Source* file to the *Dest* file on the local system. If the *Dest* file name is not specified, the **cat** command uses the name of the *Source* file. If the *Dest* file exists on the local system, it will be replaced by the *Source* file. This signal is a UNIX-specific version of the ~< signal.
~l	Pipes the output of a remote command to a local process. The command string sent to the local system is processed by the shell.

A remote pipe will only succeed if the data from the remote pipe is terminated by one of the eof characters listed in the *eofread* **tip** command variable. If one of these characters is not encountered, then the output pipe will not succeed.

When piping remote output with the ~| signal, the user will be prompted for a local command name. The user may respond with any valid command name. When prompted for the remote command, the user should append the EOF character to the end of the file being read.

This signal can be used as shown in the following example:

```
Local command? cat
List command for remote system? echo
"asdfasdfasdfasdf\04"
```

This example assumes that the character 0x4 is present in the **tip** *eofread* variable. The best way of ensuring that this character exists in the variable is to assign it in the usr's **.tiprc** file, which should reside in the user's home directory.

To accomplish this, the following command can be issued:

```
echo"eofread=\04" >> ~/.tiprc
```

~$	Pipes the output of a local process to the remote system. The command string sent to the remote system is processed by the shell.
~#	Sends a **BREAK** signal to the remote system.
~s { *Variable*=*Value* | [**!**]*BoolVariable* | **all** | *Variable***?** }	

Sets or queries the **tip** command variables.

To change the value of a non-Boolean variable, enter the variable name or abbreviation, followed by an **=** (equal sign), followed by the new value. For example, type `~s rc=^U` to change the character used to turn uppercase conversion on or off (the *raisechar* variable).

To change the value of a Boolean variable, enter the variable name or abbreviation. To reset the variable to its default value, type an **!** (exclamation point) in front of the name. For example, type `~s !ec` to reset the *echocheck* variable to its default value.

To display all variables readable by the user, specify **all** as an argument to the **~s** signal. You may also request the display of a specific variable by attaching a **?** (question mark) to the variable name. For example, type the command `~s eol?` to display the current end-of-line string (the *eol* variable).

~^Z Stops the **tip** command. The **~^Z** signal is only available with job control.

~^Y Stops the local portion of the **tip** command. The remote portion, which displays the output from the remote system, continues to run. The **~^Y** signal is only available with job control.

~? Displays a list of the escape signals.

Variables

The **tip** command uses variables that control its operation. These variables may be numeric, string, character, or Boolean values. Some of these variables can be changed by any user who can run the **tip** command. However, the following variables can be changed only by a user with root user authority: the *baudrate* variable and the *dialtimeout* variable.

Variables may be initialized at run time in the **$HOME/.tiprc** file. Additionally, you can display and set the variables while already running the **tip** command by using the **~s** escape signal.

Variables may be numeric, string, character, or Boolean values. To set a non-Boolean variable, enter the variable name or abbreviation followed by an **=** (equal sign) and the value. For example, type either `~s host=zeus` or `~s ho=zeus` to change the **host** name to `zeus`. In the **.tiprc** file, type `host=zeus` or `ho=zeus`.

To change the value of a Boolean variable, enter the variable name or abbreviation as an argument to the **~s** signal or on a line of the **.tiprc** file. To reset the variable to its default value, type an `!` (exclamation point) in front of the name. For example, type `~s !echocheck` to reset the *echocheck* variable to its default value while running the **tip** command.

Following are the common variables, their types, abbreviations, and default values.

Variable (Abbreviation)	Type	Default Value
beautify (be)	Boolean	Instructs the **tip** command to discard unprintable characters when a session is being scripted. Does not discard characters specified with the *exceptions* variable. The default setting is `on`.

Variable (Abbreviation)	Type	Default Value
baudrate (ba)	Numeric	Reflects the baud rate of the connection. Changing the value of this variable will *not* change the current baud setting of the connected tty device.
dialtimeout (dial)	Numeric	Specifies the time in seconds that the **tip** command waits for a connection when dialing a phone number. The default is 60 seconds. The **dialtimout** setting can be changed only by someone with root user authority.
echocheck (ec)	Boolean	Instructs the **tip** command to synchronize with the remote system during a file transfer by awaiting the echo of the last character transmitted before transmitting the next character. The default setting is `off`.
eofread (eofr)	String	Specifies the set of characters that signifies end-of-transmission during a remote-to-local (~< or ~t) file transfer.
eofwrite (eofw)	String	Specifies the string that is sent to indicate the end of a transmission during a local-to-remote (~> or~p) file transfer.
eol (none)	String	Specifies the string that indicates the end of a line. The **tip** command recognizes escape signals only when they follow an end-of-line string.
escape (es)	Character	Specifies the character prefix for escape signals. The default is ~ (tilde).
etimeout (et)	Numeric	Specifies the time to wait for a response when the *echocheck* variable is set `on`. If the echo is not received within the designated time, the file transfer is discontinued. The default time is 28 seconds.
exceptions (ex)	String	Specifies the set of characters that should not be discarded even when the **beautify** switch is set to `on`. The `\t\n\f\b` string is the default.
force (fo)	Character	Specifies the character that is used to force literal data transmissions during binary transfers. The `^P` character is the default. Literal data transmissions are off until the user types the character specified by the *force* variable.
framesize (fr)	Numeric	Specifies the number of bytes to buffer between files system writes when receiving files from the remote system.
host (ho)	String	Specifies the name of the remote system to which you were connected when the **tip** command was invoked. This variable cannot be changed.
halfduplex (hdx)	Boolean	Toggles Half-duplex mode. The default setting is `off`.
localecho (le)	Boolean	Toggles the Local-echo mode. The default setting is `off`.
log (none)	String	Defines the file used to log dial-outs with the **tip** command. The default file is the **/var/spool/uucp/.Admin/aculog** file. The log file can be changed only by someone with root authority.
parity (par)	String	Defines the parity for file transfers. Defaults to the following string: `no parity, 8 data bits`
phones (none)	String	Specifies the name of the user's phone file. The file can have any valid file name and must be set up in the format of the **/usr/lib/phones-file** file. The default is the **/etc/phones** file. If a file is specified with the **PHONES** environment variable, it is used in place of (not in addition to) the **/etc/phones** file.

prompt (pr)	Character	Specifies the character that indicates the end of the line on the remote host. This character is used to synchronize during data transfers. The **tip** command counts lines transferred during a file transfer, based on the number of times it receives the **prompt** character. The `\n` character is the default.
raise (ra)	Boolean	When set to `on`, instructs the **tip** command to convert all lowercase letters to uppercase before transmitting them to the remote system. The default setting is `off`.
raisechar (rc)	Character	Specifies a character that is used to toggle uppercase conversion. The `^A` character is the default.
rawftp (raw)	Boolean	If the *rawftp* variable is set to `on`, data is transmitted over the connection during a file transfer with no additional processing carried out. That is, when sending files, line-feeds are not mapped to line-feed/carriage carried out.
record (rec)	String	Specifies the name of the file in which the **tip** command records the session script. The **tip.record** file is the default. The **tip** command places the file in the user's current directory on the local system.
remote (none)	String	Specifies the name of the user's remote system definition file. The file can have any valid file name and must be set up in the format of the **/usr/lib/remote-file** file. The default is the **/etc/remote** file. If a file is specified with the **REMOTE** environment variable, it is used in place of (not in addition to) the **/etc/remote** file.
script (sc)	Boolean	When the **script** switch is set `on`, the **tip** command records everything transmitted by the remote system in a file on the local system. The file name is specified by the *record* variable. If the **beautify** switch is set to `on`, only printable ASCII characters (those between 040 and 0177) will be recorded in the script file. The *exceptions* variable specifies unprintable characters that will be recorded even if the **beautify** switch is set to `on`. The default setting for the **script** switch is `off`.
tabexpand (tab)	Boolean	Causes the **tip** command to expand tab characters to eight spaces during file transfers. The default setting is `off`.
verbose (verb)	Boolean	When the **verbose** switch is set `on`, the **tip** command prints messages while dialing, shows the current number of lines transferred during a file transfer, and displays other status information about the connection. The default setting is `on`.
SHELL (none)	String	Specifies the type of shell to use for the **~!** signal. The default value is **/usr/bin/sh** or is taken from the environment.
HOME (none)	String	Specifies the home directory to use for the **~c** signal. The default value is taken from the environment.

Examples

1. To specify a baud rate when making a direct connection, type:

   ```
   tip -300 hera
   ```

   This instructs the **tip** command to use baud rate of `300` when contacting remote system `hera`.
2. To use a modem to connect to a remote system, type:

   ```
   tip 9,343-2132
   ```

   The **tip** command connects the local system to the remote system reached by the telephone number `343-2132`, after dialing a `9` to reach an outside line.

3. To connect directly to a remote system and display the variables, type:

```
tip  -v hera
```

The **-v** flag causes the **tip** command to display the values of the variables as it reads them from the **$HOME/.tiprc** file. If the **.tiprc** file contains the following settings:

```
sc
be
rec=/home/jimk/callout
```

then output from the **-v** flag is as follows:

```
set script
set beautify
set record=/home/jimk/callout
```

Files

/usr/bin/tip	Contains the **tip** command.
/etc/locks/*	Contains lock files that prevent multiple uses of devices and multiple calls to systems.
/etc/remote	Contains system descriptions for the **tip** command. If the *remote* variable or the **REMOTE** environment variable is set, that file is used instead.
/usr/lib/remote-file	Contains sample **remote** file. If the*remote* variable or the **RECORD** environment variable is set, that file is used instead.
/etc/phones	Contains the telephone number database for the **tip** command. If the *phones* variable or the **PHONES** environment variable is set, that file is used instead.
/usr/lib/phones-file	Contains the telephone number database for the **tip** command. If the *phones* variable or the **PHONES** environment variable is set, that file is used instead.
$HOME/.tiprc	Defines initial settings for the **tip** command variables.
tip.record	Contains the **tip** command scripts. By default, the file is stored in the current directory. The user can change the file name and directory using the *record* variable.

Related Information

The **cu** command, **uucp** command.

The Communication with connected UNIX systems using the tip command in *Networks and communication management*.

---

tokstat Command

Purpose

Shows token-ring device driver and device statistics.

Syntax

tokstat [**-d -r -t**] *Device_Name*

Description

The **tokstat** command displays the statistics gathered by the specified Token-Ring device driver. The user can optionally specify that the device-specific statistics be displayed in addition to the device driver statistics. If no flags are specified, only the device driver statistics are displayed.

This command is also invoked when the **netstat** command is run with the **-v** flag. The **netstat** command does not issue any **tokstat** command flags.

If an invalid *Device_Name* is specified, the **tokstat** command produces an error message stating that it could not connect to the device.

Flags

-d Displays all the device driver statistics, including the device-specific statistics.
-r Resets all the statistics back to their initial values. This flag can only be issued by privileged users.
-t Toggles debug trace in some device drivers.

Parameters

Device_Name The name of the Token-Ring device, for example, t**ok0**.

Statistic Fields

Note: Some adapters may not support a specific statistic. The value of non-supported statistic fields is always 0.

The statistic fields displayed in the output of the **toktstat** command and their descriptions are:

Title Fields

`Device Type` Displays the description of the adapter type.
`Hardware Address` Displays the Token-Ring network address currently used by the device.
`Elapsed Time` Displays the real time period which has elapsed since the last time the statistics were reset. Part of the statistics may be reset by the device driver during error recovery when a hardware error is detected. There will be another Elapsed Time displayed in the middle of the output when this situation has occurred in order to reflect the time differences between the statistics.

Transmit Statistics Fields

`Packets` The number of packets transmitted successfully by the device.
`Bytes` The number of bytes transmitted successfully by the device.
`Interrupts` The number of transmit interrupts received by the driver from the adapter.
`Transmit Errors` The number of output errors encountered on this device. This is a counter for unsuccessful transmissions due to hardware/network errors.
`Packets Dropped` The number of packets accepted by the device driver for transmission which were not (for any reason) given to the device.
`Max Packets on S/W Transmit Queue` The maximum number of outgoing packets ever queued to the software transmit queue.
`S/W Transmit Queue Overflow` The number of outgoing packets which have overflowed the software transmit queue.
`Current S/W+H/W Transmit Queue Length` The number of pending outgoing packets on either the software transmit queue or the hardware transmit queue.

Broadcast Packets	The number of broadcast packets transmitted without any error.
Multicast Packets	The number of multicast packets transmitted without any error.
Timeout Errors	The number of unsuccessful transmissions due to adapter reported timeout errors.
Current SW Transmit Queue Length	The number of outgoing packets currently on the software transmit queue.
Current HW Transmit Queue Length	The number of outgoing packets currently on the hardware transmit queue.

Receive Statistics Fields

Packets	The number of packets received successfully by the device.
Bytes	The number of bytes received successfully by the device.
Interrupts	The number of receive interrupts received by the driver from the adapter.
Receive Errors	The number of input errors encountered on this device. This is a counter for unsuccessful reception due to hardware/network errors.
Packets Dropped	The number of packets received by the device driver from this device which were not (for any reason) given to a network demuxer.
Bad Packets	The number of bad packets received (saved) by the device driver.
Broadcast Packets	The number of broadcast packets received without error.
Multicast Packets	The number of multicast packets received without error.
Receive Congestion Errors	The number of incoming packets dropped by the hardware due to a no resource error.

General Statistics Fields

No mbuf Errors	The number of times mbufs were not available to the device driver. This usually occurs during receive operations when the driver must obtain mbuf buffers to process inbound packets. If the mbuf pool for the requested size is empty, the packet will be discarded. The **netstat -m** command can be used to confirm this.
Lobe Wire Faults	The number of times the adapter detected an open or short circuit in the lobe data path (for example, the cable is unplugged).
Abort Errors	The number of times the adapter had problems transmitting.
AC Errors	The number of times the adapter received more than one AMP (Active Monitor Present) or SMP (Standby Monitor Present) frame which had the address recognized and frame copied bits set to zero. This indicates a problem with neighbor notification. Every station learns and remembers who its Nearest Active Upstream Neighbor (NAUN) is from AMP and SMP frames. When a station reports a problem, it also reports who its NAUN is. This helps to define the *fault domain*.
Burst Errors	The number of times the adapter detected that the polarity of the signal did not switch when necessary.
Frame Copy Errors	The number of times the adapter detected that a frame with its specific address has been copied by another adapter.
Frequency Errors	The number of times the adapter detected that the frequency of the incoming signal differs from the expected frequency by more than that allowed by the IEEE 802.5 standard. Check the active monitor responsible for master clocking of the ring and compensating for frequency jitter.
Hard Errors	The number of times the adapter either transmitted or received a beacon MAC frame.
Internal Errors	The number of times the adapter had an internal error.

Line Errors	The number of times the adapter detected an invalid character in a frame or token.
Lost Frame Errors	The number of times the adapter transmitted a frame and failed to receive it back.
Only Station	The number of times the adapter sensed that it is the only adapter on the ring.
Token Errors	The number of times the adapter, acting as an active monitor, detected that the token got lost. This may be due to ring reconfiguration. If this occurs often, check to see if other soft errors indicate a specific problem.
Remove Received	The number of times the adapter received a Remove Ring Station MAC frame request.
Ring Recovered	The number of times the ring is purged and recovered back into a normal operating state.
Signal Loss Errors	The number of times the adapter detected the absence of a receive signal.
Soft Errors	The number of times the adapter detected a soft error (recoverable by the MAC layer protocols).
Transmit Beacon Errors	The number of times the adapter transmitted a beacon frame.
Driver Flags	The device driver internal status flags currently turned on.

Device Specific Statistics Fields

This part of the display may be different for each type of adapter. It may contain adapter-specific information and some extended statistics that were not included in the generic statistics. Some adapters may not have any device-specific statistics. Some fields that may be listed in this section are:

ARI/FCI Errors	ARI/FCI mismatch is also referred to as receiver congestion. If an adapter gets an address match on a frame going by on the ring, Address Recognized Indication(ARI), and has no place into which to copy the frame, Frame Copied Indication(FCI), an ARI/FCI mismatch has occurred. The adapter will turn on the ARI bits but will not turn on the FCI bits in the FS byte at the end of the frame as it goes by. In other words, the adapter saw a frame that was to be received but, could not receive it because the receive buffers have been depleted. Two seconds later the adapter will send a Report Soft Error MAC frame indicating a receiver congestion error.
DMA Bus Errors	The number of times the adapter completed a DMA transfer and detected a bus error.
DMA Parity Errors	The number of times the adapter completed a DMA transfer and detected a parity error.
Receive Overruns	The number of times the adapter receive FIFO was full when the adapter tried to receive a frame.
Receive Underruns	The number of times the adapter transmit FIFO was empty before the end of frame symbol was detected.
Number of read log commands issued	The number of times an adapter error counter overruns (reached 255) and the device driver issues a read log command to read (and reset) the error counters.

Examples

1. To display the device driver statistics for **tok0**, enter:

   ```
   tokstat tok0
   ```

   This produces the following output:

   ```
   TOKEN-RING STATISTICS (tok0) :
   Device Type: Token-Ring High-Performance Adapter (8fc8)
   Hardware Address: 10:00:5a:4f:26:c1
   Elapsed Time: 0 days 0 hours 8 minutes 33 seconds
   Transmit Statistics:                 Receive Statistics:
   --------------------                 -------------------
   Packets: 191                         Packets: 8342
   Bytes: 17081                         Bytes: 763227
   Interrupts: 156                      Interrupts: 8159
   Transmit Errors: 0                   Receive Errors: 0
   Packets Dropped: 0                   Packets Dropped: 0
   Max Packets on S/W Transmit Queue: 17 Bad Packets: 0
   S/W Transmit Queue Overflow: 0
   Current S/W+H/W Transmit Queue Length: 0

   Broadcast Packets: 1                 Broadcast Packets: 8023
   Multicast Packets: 0                 Multicast Packets: 0
   Timeout Errors: 0                    Receive Congestion Errors: 0
   Current SW Transmit Queue Length: 0
   Current HW Transmit Queue Length: 0

   General Statistics:
   -------------------
   No mbuf Errors: 0                    Lobe Wire Faults: 0
   Abort Errors: 0                      AC Errors: 0
   Burst Errors: 0                      Frame Copy Errors: 0
   Frequency Errors: 0                  Hard Errors: 0
   Internal Errors: 0                   Line Errors: 0
   Lost Frame Errors: 0                 Only Station: 0
   Token Errors: 0                      Remove Received: 0
   Ring Recovered: 0                    Signal Loss Errors: 0
   Soft Errors: 0                       Transmit Beacon Errors: 0
   Driver Flags: Up Broadcast Running
      AlternateAddress ReceiveFunctionalAddr
   ```

2. To display the token-ring device driver statistics and the Token-Ring device-specific statistics for **tok0**, enter:

   ```
   tokstat -d tok0
   ```

   This produces the following output:

   ```
   TOKEN-RING STATISTICS (tok0) :
   Device Type: Token-Ring High-Performance Adapter (8fc8)
   Hardware Address: 10:00:5a:4f:26:c1
   Elapsed Time: 0 days 2 hours 48 minutes 38 seconds

   Transmit Statistics:                 Receive Statistics:
   --------------------                 -------------------
   Packets: 389                         Packets: 153216
   Bytes: 42270                         Bytes: 14583150
   Interrupts: 354                      Interrupts: 151025
   Transmit Errors: 0                   Receive Errors: 0
   Packets Dropped: 0                   Packets Dropped: 0
   Max Packets on S/W Transmit Queue:17 Bad Packets: 0
   S/W Transmit Queue Overflow: 0
   Current S/W+H/W Transmit Queue Length: 0

   Broadcast Packets: 1                 Broadcast Packets: 152642
   Multicast Packets: 0                 Multicast Packets: 0
   Timeout Errors: 0                    Receive Congestion Errors: 0
   Current SW Transmit Queue Length: 0
   Current HW Transmit Queue Length: 0
   ```

```
General Statistics:
-------------------
No mbuf Errors: 0                   Lobe Wire Faults: 0
Abort Errors: 0                     AC Errors: 0
Burst Errors: 0                     Frame Copy Errors: 0
Frequency Errors: 0                 Hard Errors: 0
Internal Errors: 0                  Line Errors: 0
Lost Frame Errors: 0                Only Station: 0
Token Errors: 0                     Remove Received: 0
Ring Recovered: 0                   Signal Loss Errors: 0
Soft Errors: 0                      Transmit Beacon Errors: 0
Driver Flags: Up Broadcast Running
   AlternateAddress ReceiveFunctionalAddr

Token-Ring High-Performance Adapter (8fc8) Specific Statistics:
---------------------------------------------------------------
DMA Bus Errors: 0                      DMA Parity Errors: 0
ARI/FCI Errors: 0
```

Related Information

The **atmstat** command, **entstat** command, **fddistat** command, **netstat** command.

---

topas Command

Purpose

Reports selected local and remote system statistics.

Syntax

topas [**-d** *number_of_monitored_hot_disks*] [**-h**] [**-i** *monitoring_interval_in_seconds*] [**-n** *number_of_monitored_hot_network_interfaces*] [**-p** *number_of_monitored_hot_processes*] [**-w** *number_of_monitored_hot_WLM classes*] [**-c** *number_of_monitored_hot_CPUs*][**-l** *remote_polling_interval*] [**-U** *username_owned_processes*] | [**-C** [**-o** *field* **=** *value*,...] **-D** | **-L** |**-P** | **-W**] [**-m**]

Description

The **topas** command reports selected statistics about the activity on the local system. The command uses the curses library to display its output in a format suitable for viewing on an 80x25 character-based display or in a window of at least the same size on a graphical display. The **topas** command requires the **bos.perf.tools** and **perfagent.tools** filesets to be installed on the system.

The **topas** command can also report a limited set of performance metrics from remote AIX partitions that belong to the same hardware platform. This support is described in the Cross-Partition View section.

If the **topas** command is invoked without flags, it runs as if invoked with the following command line:

```
topas -d20 -i2 -n20 -p20 -w20 -c20
```

The program extracts statistics from the system with an interval specified by the *monitoring_interval_in_seconds* argument. The default output, as shown below, consists of two fixed parts and a variable section. The top two lines at the left of the display show the name of the system the **topas** command runs on, the date and time of the last observation, and the monitoring interval.

The second fixed part fills the rightmost 25 positions of the display. It contains five subsections of statistics, as follows:

EVENTS/QUEUES Displays the per-second frequency of selected system-global events and the average size of the thread run and wait queues:

Cswitch
The number of context switches per second over the monitoring interval.

Syscalls
The total number of system calls per second executed over the monitoring interval.

Reads The number of read system calls per second executed over the monitoring interval.

Writes The number of write system calls per second executed over the monitoring interval.

Forks The number of fork system calls per second executed over the monitoring interval.

Execs The number of exec system calls per second executed over the monitoring interval.

Runqueue
The average number of threads that were ready to run but were waiting for a processor to become available.

Waitqueue
The average number of threads that were waiting for paging to complete.

FILE/TTY Displays the per-second frequency of selected file and tty statistics.

Readch
The number of bytes read per second through the **read** system call over the monitoring interval.

Writech
The number of bytes written per second through the **write** system call over the monitoring interval.

Rawin The number of raw bytes read per second from TTYs over the monitoring interval.

Ttyout The number of bytes written to TTYs per second over the monitoring interval.

Igets The number of calls per second to the inode lookup routines over the monitoring interval.

Namei The number of calls per second to the pathname lookup routines over the monitoring interval.

Dirblk The number of directory blocks scanned per second by the directory search routine over the monitoring interval.

PAGING Displays the per-second frequency of paging statistics.

Faults Total number of page faults taken per second over the monitoring interval. This includes page faults that do not cause paging activity.

Steals Physical memory 4K frames stolen per second by the virtual memory manager over the monitoring interval.

PgspIn Number of 4K pages read from paging space per second over the monitoring interval.

PgspOut
Number of 4K pages written to paging space per second over the monitoring interval.

PageIn Number of 4K pages read per second over the monitoring interval. This includes paging activity associated with reading from file systems. Subtract **PgspIn** from this value to get the number of 4K pages read from file systems per second over the monitoring interval.

PageOut
Number of 4K pages written per second over the monitoring interval. This includes paging activity associated with writing to file systems. Subtract **PgspOut** from this value to get the number of 4K pages written to file systems per second over the monitoring interval.

Sios The number of I/O requests per second issued by the virtual memory manager over the monitoring interval.

MEMORY Displays the real memory size and the distribution of memory in use.

Real,MB
The size of real memory in megabytes.

% Comp
The percentage of real memory currently allocated to computational page frames. Computational page frames are generally those that are backed by paging space.

% Noncomp
The percentage of real memory currently allocated to non-computational frames. Non-computational page frames are generally those that are backed by file space, either data files, executable files, or shared library files.

% Client
The percentage of real memory currently allocated to cache remotely mounted files.

PAGING SPACE Display size and utilization of paging space.

Size,MB
The sum of all paging spaces on the system, in megabytes.

% Used
The percentage of total paging space currently in use.

% Free The percentage of total paging space currently free.

NFS Display NFS stats in calls per second

- Server V2 calls/sec
- Client V2 calls/sec
- Server V3 calls/sec
- Client V3 calls/sec

The variable part of the **topas** display can have one, two, three, four, or five subsections. If more than one subsection displays, they are always shown in the following order:

- CPU
- Network Interfaces

- Physical Disks
- WorkLoad Management Classes
- Processes

When the **topas** command is started, it displays all subsections for which hot entities are monitored. The exception to this is the WorkLoad Management (WLM) Classes subsection, which is displayed only when WLM is active.

CPU Utilization This subsection displays a bar chart showing cumulative CPU usage. If more than one CPU exists, a list of CPUs can be displayed by pressing the **c** key *twice*. Pressing the **c** key only once will turn this subsection off. The following fields are displayed by both formats:

User% This shows the percent of CPU used by programs executing in user mode. (Default sorted by User%)

Kern% This shows the percent of CPU used by programs executing in kernel mode.

Wait% This shows the percent of time spent waiting for IO.

Idle% This shows the percent of time the CPU(s) is idle.

Physc Number of physical processors consumed. Displayed only if the partition is running with shared processor.

%Entc Percentage of entitled capacity consumed. Displayed only if the partition is running with shared processor.

When this subsection first displays the list of hot CPUs, the list is sorted by the User% field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column.

Network Interfaces This subsection displays a list of hot network interfaces. The maximum number of interfaces displayed is the number of hot interfaces being monitored, as specified with the **-n** flag. A smaller number of interfaces will be displayed if other subsections are also being displayed. Pressing the **n** key turns off this subsection. Pressing the **n** key again shows a one-line report summary of the activity for all network interfaces. Both reports display the following fields:

Interf The name of the network interface.

KBPS The total throughput in megabytes per second over the monitoring interval. This field is the sum of kilobytes received and kilobytes sent per second.

I-Pack The number of data packets received per second over the monitoring interval.

O-Pack
The number of data packets sent per second over the monitoring interval.

KB-In The number of kilobytes received per second over the monitoring interval.

KB-Out
The number of kilobytes sent per second over the monitoring interval.

When this subsection first displays the list of hot network interfaces, the list is sorted by the KBPS field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 16 network adapters.

Physical Disks

This subsection displays a list of hot physical disks. The maximum number of physical disks displayed is the number of hot physical disks being monitored as specified with the **-d** flag. A smaller number of physical disks will be displayed if other subsections are also being displayed. Pressing the **d** key turns off this subsection. Pressing the **d** key again shows a one-line report summary of the activity for all physical disks. Both reports display the following fields:

Disk The name of the physical disk.

Busy% Indicates the percentage of time the physical disk was active (bandwidth utilization for the drive).

KBPS The number of kilobytes read and written per second over the monitoring interval. This field is the sum of **KB-Read** and **KB-Read**.

TPS The number of transfers per second that were issued to the physical disk. A transfer is an I/O request to the physical disk. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.

KB-Read
The number of kilobytes read per second from the physical disk.

K -Writ The number of kilobytes written per second to the physical disk.

When this subsection first displays the list of hot physical disks, the list is sorted by the KBPS field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column. Sorting is only valid for up to 128 physical disks.

WLM Classes

This subsection displays a list of hot WorkLoad Management (WLM) Classes. The maximum number of WLM classes displayed is the number of hot WLM classes being monitored as specified with the **-w** flag. A smaller number of classes will be displayed if other subsections are also being displayed. Pressing the **w** key turns off this subsection. The following fields are displayed for each class:

% CPU Utilization
The average CPU utilization of the WLM class over the monitoring interval.

% Mem Utilization
The average memory utilization of the WLM class over the monitoring interval.

% Blk I/O
The average percent of Block I/O of the WLM class over the monitoring interval.

When this subsection first displays the list of hot WLM classes, the list will be sorted by the CPU% field. However, the list can be sorted by the other fields by moving the cursor to the top of the desired column.

Processes

This subsection displays a list of hot processes. The maximum number of processes displayed is the number of hot processes being monitored as specified with the **-p** flag. A smaller number of processes will be displayed if other subsections are also being displayed. Pressing the **p** key turns off this subsection. The processes are sorted by their CPU usage over the monitoring interval. The following fields are displayed for each process:

Name
The name of the executable program executing in the process. The name is stripped of any pathname and argument information and truncated to 9 characters in length.

Process ID
The process ID of the process.

% CPU Utilization
The average CPU utilization of the process over the monitoring interval. The first time a process is shown, this value is the average CPU utilization over the lifetime of the process.

Paging Space Used
The size of the paging space allocated to this process. This can be considered an expression of the footprint of the process but does not include the memory used to keep the executable program and any shared libraries it may depend on.

Process Owner (if the WLM section is off)
The user name of the user who owns the process.

WorkLoad Management (WLM) Class (if the WLM section is on)
The WLM class to which the process belongs.

Cross-Partition View

This panel displays metrics similar to the **lparstat** command for all the AIX partitions it can identify as belonging to the same hardware platform. Dedicated and shared partitions are displayed in separate sections with appropriate metrics. The top section represents aggregated data from the partition set to show overall partition, memory, and processor activity.

Remote enablement for this panel to collect from other partitions requires that latest updates to the perfagent.tools and bos.perf.tools used to support this function. For earlier versions of AIX, the topas command also collects remote data from partitions that have the Performance Aide product (perfagent.server) installed. **topas -C** may not be able to locate partitions residing on other subnets. To circumvent this, create a $HOME/**Rsi.hosts** file containing the fully qualified hostnames for each partition (including domains), one host per line.

The following metrics display in the initial Cross-Partition panel. Additional metrics with full descriptive labels can be displayed by using the key toggles identified in the Additional Cross-Partition Panel Subcommands section:

```
Partition totals
   Shr        Number of shared partitions
   Ded        Number of dedicated partitions
Memory (in GB)
   Mon        Monitored partitions total memory
   Avl        Memory available to partition set
   InUse      Memory in use on monitored partitions
Processor
   Shr        Number of shared processors
   Ded        Number of dedicated processors
   PSz        Active physical CPUs in the shared processor pool being used by this LPAR
   APP        Available physical processors in the shared pool
   Shr_PhysB  Shared Physical Busy
   Ded_PhysB  Dedicated Physical Busy
```

```
Individual partition data
   Host       Hostname
   OS         Operating system level
   M          Mode
                     For shared partitions
                            C - SMT enabled & capped
                            c - SMT disabled & capped
                            U - SMT enabled & uncapped
                            u - SMT disabled & uncapped
                     For dedicated partitions
                            S - SMT enabled
                            ' ' (blank) - SMT enabled
   Mem        Total memory in GB.
   InU        Memory in use in GB.
   Lp         Number of logical processors
   Us         Percentage of CPU used by programs executing in user mode.
   Sy         Percentage of CPU used by programs executing in kernel mode.
   Wa         Percentage of time spent waiting for IO.
   Id         Percentage of time the CPU(s) is idle.
   Ded_PhysB  Dedicated Physical Busy
   Ent        Entitlement granted (shared-only)
   %Entc      Percent Entitlement consumed (shared-only)
   Vcsw       Virtual context switches average per second (shared-only)
   PhI        Phantom interrupts average per second (shared-only)
```

Implementation Specifics

Disks and network adapters added after starting **topas** or any other SPMI consumer will not be reflected in **topas**. You must stop **topas** and all clients that use SPMI and then restart after the changes to disks and network adapters are made.

Sample Default Output

The following is an example of the display generated by the **topas** command:

```
Topas Monitor for host:    niller              EVENTS/QUEUES    FILE/TTY
Mon Mar 13 15:56:32 2000   Interval:  2        Cswitch     113  Readch  1853576
                                               Syscall    2510  Writech   49883
CPU   User%   Kern%  Wait%  Idle%             Reads       466  Rawin         0
cpu0    7.0     4.0    0.0   89.0             Writes       12  Ttyout      706
cpu1    1.0     8.0    0.0   91.0             Forks         0  Igets         0
cpu2    0.0     0.0    0.0  100.0             Execs         0  Namei         0
                                               Runqueue    0.0  Dirblk        0
                                               Waitqueue   0.0
Interf   KBPS   I-Pack  O-Pack   KB-In  KB-Out
lo0      100.4    45.7    45.7    50.2    50.2  PAGING           MEMORY
tr0        2.0     4.4     3.4     1.4     0.6  Faults        1  Real,MB     255
                                               Steals        0  % Comp     81.0
Disk    Busy%     KBPS     TPS KB-Read KB-Writ  PgspIn        0  % Noncomp  19.0
hdisk0    0.0      0.0     0.0     0.0     0.0  PgspOut       0  % Client    3.0
hdisk1    0.0      0.0     0.0     0.0     0.0  PageIn        0
                                               PageOut       0  PAGING SPACE
WLM-Class (Active)  CPU%    Mem%     Disk%     Sios          0  Size,MB       0
System                 8      41        12                      % Used
Shared                 1      24         9                      % Free
                                               NFS      calls/sec
Name     PID      CPU%   PgSP  Class           ServerV2    0   Press:
topas    (35242)  3.0    0.3   System          ClientV2    0   "h" for help.
X        (3622)   1.4   44.4   System          ServerV3    0   "q" to quit.
notes    (25306)  1.3  123.3   System          ClientV3    0
```

Sample Full Screen Process Output

```
Topas Monitor for host:    mothra      Interval:   2    Wed Nov  8 12:27:34 2000
                             DATA  TEXT  PAGE               PGFAULTS
USER       PID  PPID PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root      1806     0  37 41    16  3374    16   13:25  1.0    0    0 gil
```

```
root      1032     0  16 41     3  3374     3    0:00  0.0    0    0 lrud
root      1290     0  60 41     4  3374     4    0:02  0.0    0    0 xmgc
root      1548     0  36 41     4  3374     4    0:26  0.0    0    0 netm
root         1     0  60 20   197     9   180    0:24  0.0    0    0 init
root      2064     0  16 41     4  3374     4    0:04  0.0    0    0 wlmsched
root      2698     1  60 20    14     2    14    0:00  0.0    0    0 shlap
root      3144     1  60 20    40     1    36    5:19  0.0    0    0 syncd
root      3362     0  60 20     4  3374     4    0:00  0.0    0    0 lvmbb
root      3666     1  60 20   135    23   123    0:00  0.0    0    0 errdemon
root      3982     0  60 20     4  3374     4    0:01  0.0    0    0 rtcmd
root      4644     1  17 20     6  3374     6    0:00  0.0    0    0 dog
root      4912     1  60 20   106    13    85    0:00  0.0    0    0 srcmstr
root      5202  4912  60 20    94     8    84    0:01  0.0    0    0 syslogd
root      5426  4912  60 20   195    76   181    0:12  0.0    0    0 sendmail
root      5678  4912  60 20   161    11   147    0:01  0.0    0    0 portmap
root      5934  4912  60 20   103    11    88    0:00  0.0    0    0 inetd
root      6192  4912  60 20   217    61   188    0:21  0.0    0    0 snmpd
root      6450  4912  60 20   137    10   116    0:00  0.0    0    0 dpid2
root      6708  4912  60 20   157    29   139    0:06  0.0    0    0 hostmibd
root         0     0  16 41     3  3374     3    7:08  0.0    0    0
root      6990     1  60 20   106    10    86    0:06  0.0    0    0 cron
```

Sample Full-Screen WorkLoad Management Classes Output

```
Topas Monitor for host:    mothra      Interval:   2    Wed Nov  8 12:30:54 2000
WLM-Class (Active)              CPU%      Mem%     Disk-I/O%
System                           0         0           0
Shared                           0         0           0
Default                          0         0           0
Unmanaged                        0         0           0
Unclassified                     0         0           0
```

```
==============================================================================
                              DATA  TEXT  PAGE              PGFAULTS
USER       PID  PPID PRI NI   RES   RES SPACE    TIME CPU%  I/O  OTH COMMAND
root         1     0 108 20   197     9   180    0:24  0.0    0    0 init
root      1032     0  16 41     3  3374     3    0:00  0.0    0    0 lrud
root      1290     0  60 41     4  3374     4    0:02  0.0    0    0 xmgc
root      1548     0  36 41     4  3374     4    0:26  0.0    0    0 netm
root      1806     0  37 41    16  3374    16   13:25  0.0    0    0 gil
root      2064     0  16 41     4  3374     4    0:04  0.0    0    0 wlmsched
root      2698     1 108 20    14     2    14    0:00  0.0    0    0 shlap
root      3144     1 108 20    40     1    36    5:19  0.0    0    0 syncd
root      3362     0 108 20     4  3374     4    0:00  0.0    0    0 lvmbb
root      3666     1 108 20   135    23   123    0:00  0.0    0    0 errdemon
root      3982     0 108 20     4  3374     4    0:01  0.0    0    0 rtcmd
```

Sample Full-Screen Cross-Partition Output

```
Topas CEC Monitor             Interval:  10           Wed Mar  6 14:30:10 2005
Partitions     Memory (GB)                Processors
Shr:  4        Mon:  24  InUse:  14       Mon:  8  PSz:  4  Shr_PhysB:  1.7
Ded:  4        Avl:  24                   Avl:  8  APP:  4  Ded_PhysB:  4.1

Host         OS  M Mem InU Lp  Us Sy Wa Id  PhysB  Ent  %EntC Vcsw PhI
---------------------------------shared-----------------------------------------
ptools1      A53 u 1.1 0.4  4  15  3  0 82   1.30  0.50  22.0  200   5
ptools5      A53 U  12  10  1  12  3  0 85   0.20  0.25   0.3  121   3
ptools3      A53 C 5.0 2.6  1  10  1  0 89   0.15  0.25   0.3   52   2
ptools7      A53 c 2.0 0.4  1   0  1  0 99   0.05  0.10   0.3  112   2
```

```
-------------------------------dedicated---------------------------------------
ptools4      A53 S 0.6 0.3  2  12  3  0 85   0.60
ptools6      A52   1.1 0.1  1  11  7  0 82   0.50
ptools8      A52   1.1 0.1  1  11  7  0 82   0.50
ptools2      A52   1.1 0.1  1  11  7  0 82   0.50
```

Flags

-c Specifies the number of hot CPUs to be monitored. This is also the maximum number of CPUs displayed when enough room is available on the screen. If this number exceeds the number of CPUs available, only the installed CPUs will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no CPU information is monitored.

-C Displays the Cross-Partition panel. The **topas** command collects a set of metrics from AIX partitions running on the same hardware platform. The metrics are similar to those collected by the **lparstat** command. Dedicated and shared partitions are displayed, and a set of aggregated values provide an overview of the entire hardware systems partition set. Certain values only available from the HMC platform may be set via the **line** command if an HMC connection is not available:

- -o availmem = [total amount of memory allocated to all partitions, in GB]
- -o unavailmem = [total amount of memory unallocated from the HMC, in GB]
- -o availcpu = [total number of physical processors allocated for all partitions]
- -o unavailcpu = [total number of physical processors unallocated from the HMC]
- -o poolsize = [defined Pool Size required if HMC Processor Utilization Authority restricts access]
- -o partitions = [number of partitions defined on the HMC]
- -o reconfig = [number of seconds between checking for HMC configuration changes. Allowed values are 30, 60, 90, 120, 180, 240, 300 seconds. The default is 60 seconds]

-D Displays the Disk Metrics display. The display reports disk service times, disk queuing metrics, and disk throughput. The following metrics are reported:

AQD Average number of requests waiting to be sent to disk.

AQW Average queue wait per request reported in milliseconds.

ART Indicates the average time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is milliseconds.

AWT Indicates the average time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is milliseconds.

MRT Indicates the maximum time to receive a response from the hosting server for the read request sent. The suffix indicates the unit of time. The default time unit is milliseconds.

MWT Indicates the maximum time to receive a response from the hosting server for the write request sent. The suffix indicates the unit of time. The default time unit is milliseconds.

-d Specifies the number of disks to be monitored. This is also the maximum number of disks displayed when enough room is available on the screen. When this number exceeds the number of disks installed, only the installed disks will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no disk information is monitored.

-h Displays help information in the following format:

```
usage:  topas [-d number-of-monitored-hot-disks]
              [-h]
              [-i monitoring-interval_in_seconds]
              [-n number-of-monitored-hot-network-interfaces]
              [-p number-of-monitored-hot-processes]
              [-w number-of-monitored-hot-WLM classes]
              [-c number-of-monitored-hot-CPUs]
              [-U username_owned_processes]
              [-D|-P|-W|-L]
              [-m]
```

-i Sets the monitoring interval in seconds. The default is 2 seconds.

-I *remote_polling_interval* For cross-partition display, sets the sampling interval to collect data from remote partitions. The default is 10 seconds. Values of 10, 15, 30, 60 and 120 seconds are allowed.

-L Displays the logical partition display. This display reports similar data to what is provided to **mpstat** and **lparstat**.

-m Displays in monochrome mode (no colors).

-n Specifies the number of hot network interfaces to be monitored. This is also the maximum number of network interfaces displayed when enough room is available on the screen. When this number exceeds the number of network interfaces installed, only the installed network interfaces will be monitored and displayed. If this argument is omitted, a default of 2 is assumed. If a value of 0 (zero) is specified, no network information is monitored.

-P Similar to the **ps** command, the **-P** flag displays the full-screen process display. This display shows a list of the busiest processes, similar to the process subsection on the default display, only with more columns showing more metrics per process. This list can be sorted by any column. Following are the metrics displayed.

USER The login name of the process owner.

PID The process ID of the process.

PPID The process ID of the parent process.

PRI The priority of the process or kernel thread; higher numbers mean lower priority.

NI The nice value; used in calculating priority for the sched other policy.

Data RSS
The real-memory data (resident set) size of the process (4KB pages).

TEXT RSS
The real-memory text (resident set) size of the process (4KB pages).

PAGE SPACE
Virtual working set size used by process (4KB pages). **Note:** The true paging space allocations per process are not available using the **topas** command. Refer to the **svmon** command for more detailed reports.

TIME The total execution time for the process.

CPU% The percentage of CPU usage.

PGFAULTS
Shows the number of I/O and other page faults.

COMMAND
Contains the command name. Truncates the command name to 9 characters.

-p Specifies the number of hot processes to be monitored. This is also the maximum number of processes shown when enough room is available on the screen. If this argument is omitted, a default of `20` is assumed. If a value of `0` is specified, no process information will be monitored. Retrieval of process information constitutes the majority of the **topas** overhead. If process information is not required, always use this option to specify that you do not want process information.

-U *username* Shows *username* owned processes with the **-P** flag. Only processes owned by *username* will be shown in the `All Process Display`.

-W Displays the full-screen WLM class display, which is a split display. The top part of the display shows a list of hot WLM classes, similar to the WLM classes subsection on the default display, but with enough space available to display the full class names. This list can be sorted on any column.

The bottom part of the display shows a list of busiest processes, similar to the full screen process display, but only displays processes belonging to one WLM class (selected with the **f** key).

-w Specifies the number of hot WorkLoad Management (WLM) classes to be monitored. This is also the maximum number of WLM classes displayed when enough room is available on the screen. If this number exceeds the number of WLM classes installed, only the installed WLM classes will be monitored and displayed. If this argument is omitted, a default of `2` is assumed. If a value of `0` (zero) is specified, no WLM class information is monitored.

General Subcommands

While **topas** is running, it accepts one-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to the action requested.

a The **a** key shows all of the variable subsections being monitored (CPU, network, disk, WLM, and process). Pressing the **a** key always returns the **topas** command to the initial main display.

c The **c** key toggles the CPU subsection between the cumulative report, off, and a list of the busiest CPUs. The number of busiest CPUs displayed will depend upon the space available on the screen.

C The uppercase C key activates the Cross-Partition panel, or if currently active, resets the panel to display the global summary, dedicated, and shared sections. See the Additional Cross-Partition Panel Subcommands section below for options specific to this panel.

d The **d** key toggles the disk subsection between a list of busiest disks, off, and the report on the total disk activity of the system. The number of busiest disks displayed will depend upon the space available on the screen.

D The uppercase **D** key replaces the current display with the Disk Metric display. This display offers additional information about disk access times and disk queuing. When the **D** key is pressed again, it toggles back to the default main screen.

f Moving the cursor over a WLM class and pressing the **f** key displays the list of top processes in the class at the bottom of the WLM screen. This key is valid only when topas is in the full-screen WLM display (by using the **W** key or the **-W** flag).

h Show the help screen.

H Shows the help screen for the local panel, if available.

L The uppercase L key replaces the current display with the logical partition display; LPAR, Micro-Partitioning, and simultaneous multi-threading metrics similar to what **lparstat** and **mpstat** provide are displayed.

n The **n** key toggles the network interfaces subsection between a list of busiest interfaces, off, and the report on the total network activity of the system. The number of busiest interfaces displayed will depend upon the space available on the screen.

p The **p** key toggles the hot processes subsection on and off. The number of busiest processes displayed will depend upon the space available on the screen.

P	The uppercase **P** key replaces the default display with the full-screen process display. This display provides more detailed information about processes running on the system than the process section of the main display. When the **P** key is pressed again, it toggles back to the default main display.
q	Quit the program.
r	Refresh the display.
w	The **w** key toggles the WorkLoad Management (WLM) classes subsection on and off. The number of busiest WLM classes displayed will depend upon the space available on the screen.
W	The uppercase **W** key replaces the default display with the full-screen WLM class display. This display provides more detailed information about WLM classes and processes assigned to classes. When the **W** key is pressed again, it toggles back to the default main display.
Arrow and Tab keys	Subsections from the main display such as the CPU, Network, Disk, WLM Classes, and the full-screen WLM and Process displays can be sorted by different criteria. Positioning the cursor over a column activates sorting on that column. The entries are always sorted from highest to lowest value. The cursor can be moved by using the **Tab** key or the arrow keys. Sorting is only valid for 128 disks and 16 network adapters.

Additional Cross-Partition Panel Subcommands

When the **topas** Cross-Partition panel is active, it will accept the following additional one-character subcommands. Each time the monitoring interval elapses, the program checks for one of the following subcommands and responds to any requested action.

d	The d key toggles the dedicated partition section on and off.
g	The g key toggles the top global section of the panel between brief listing, detailed listing, and off.
r	The r key forces **topas** to search the for HMC configuration changes if a connection is available. This includes the discovery of new partitions, processors, or memory allocations.
s	The s key toggles the shared partition section on and off.

Examples

1. To display up to twenty ″hot″ disks every five seconds and omit network interface, WLM classes, and process information, type:

   ```
   topas -i5 -n0 -p0 -w0
   ```

2. To display the five most active processes and up to twenty most active WLM classes (which is the default when omitting the **-w** flag) but no network or disk information, type:

   ```
   topas -p5 -n0 -d0
   ```

3. To run the program with default options, type:

   ```
   topas
   ```

4. To go directly to the process display, type:

   ```
   topas -P
   ```

5. To go directly to the WLM classes display, type:

   ```
   topas -W
   ```

6. To go directly to the logical partition display, type:

   ```
   topas -L
   ```

7. To go directly to the disk metric display, type:

   ```
   topas -D
   ```

Files

/usr/bin/topas	Contains the **topas** command.

Related Information

The **iostat** command, **vmstat** command, **wlmstat** command, **sar** command, **ps** command, **mpstat** command, **lparstat** command.

System Performance Measurement Interface in the *Performance Toolbox Version 2 and 3 for AIX: Guide and Reference*

The **lchmon** sample program that ships with **perfagent.server**.

---

topasout Command

Purpose

Tabulates the contents of an **xmwlm** recording file.

Syntax

topasout [**-c** | **-s**] [**-m** *type*] *recording_file*

Description

The **topasout** command provides post-processing of recordings made by **xmwlm**. It can generate reports in ASCII or spreadsheet format.

Flags

-c	Specifies that ptxtab should format the output files as comma-separated ASCII. Each line in the output files contains one time stamp and one observation. Both fields are preceded by a label that describes the fields.
-s	Specifies that **topasout** should format the output files in a format suitable for input to spreadsheet programs. If this flag is specified, it is always assumed that the **-r** flag is also specified.
-m *type*	By default, the post-processor only outputs mean values. Other recorded values and the full set are available via other options (min, max, mean, stdev, set, exp).

Examples

1. The following is an example of **topasout** comma-separated output format:

```
#Monitor: Nice Monitor --- hostname: nchris
Time="1994/01/07   15:36:03",   PagSp/%totalused=27.82
Time="1994/01/07   15:36:03",   PagSp/%totalfree=72.18
Time="1994/01/07   15:36:03",   Mem/Virt/pagein=8
Time="1994/01/07   15:36:03",   Mem/Virt/pageout=20
Time="1994/01/07   15:36:07",   PagSp/%totalused=27.82
Time="1994/01/07   15:36:07",   PagSp/%totalfree=72.18
Time="1994/01/07   15:36:07",   Mem/Virt/pagein=7
Time="1994/01/07   15:36:07",   Mem/Virt/pageout=17
```

2. The following is an example of **topasout** spreadsheet output format:

```
#Monitor: Nice Monitor --- hostname: nchris
"Timestamp"             "PagSp/%totalused" "PagSp/%totalfree" "Mem/Virt/pagein "Mem/Virt/pageout"
"1994/01/07  15:36:03"   27.8               72.2               8                20
"1994/01/07  15:36:07"   27.8               72.2               7                17
"1994/01/07  15:36:11"   27.8               72.2               3                283
"1994/01/07  15:36:15"   27.8               72.2               28               48
"1994/01/07  15:36:19"   28.2               71.8               56               41
```

```
"1994/01/07  15:36:23"   29.5              70.5              29              38
"1994/01/07  15:36:27"   31.5              68.5              0               62
"1994/01/07  15:36:31"   32.4              67.6              70              1
"1994/01/07  15:36:35"   32.6              67.4              73              32
"1994/01/07  15:37:04"   28.0              72.0              15              0
```

Location

/usr/bin/topasout

Files

/usr/bin/topasout	Contains the **topasout** command. The **topasout** command is included in the perfagent.tools fileset.

Related Information

The **topas** command and **xmwlm** command.

---

topsvcs Command

Purpose

Starts or restarts topology services on a cluster node.

Syntax

topsvcs

Description

Use **topsvcs** script to start the operation of topology services for a cluster.

The **topsvcs** script is not normally executed from the command line. It is normally called by the **topsvcsctrl** control script, which is in turn called by the HACMP/ES startup process.

The **topsvcs** script issues these commands:

```
no -o nonlocsrcroute=1
no -o ipsrcroutesend=1
no -o ipsrcrouterecv =1
no -o ipsrcrouteforward=1
```

These commands enable IP source routing. Do not change this setting, because the topology services subsystem requires this setting to work properly. If you change the setting, the topology services subsystem and a number of other subsystems that depend on it will no longer operate properly.

Flags

–s Instructs the topology services daemon to reject messages that are apparently delayed.

–d Instructs the topology services daemon not to reject messages that are apparently delayed (this is the default).

Security

You must have **root** privilege to run this command.

Exit Status

0 Indicates the successful completion of the command.

1 Indicates the command was unsuccessful.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX-domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to **/var/ha/soc/hats/server_socket.***partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in an HACMP environment only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

To instruct the topology services daemon on the local node to start discarding apparently delayed messages, enter:

```
export HA_SYSPAR_NAME=partition1

/usr/sbin/rsct/bin/hatsoptions -s
```

Location

/usr/sbin/rsct/bin/topsvcs Contains the **topsvcs** script

Files

/var/ha/soc/hats/server_socket.*partition name*

Related Information

Commands: **lssrc**, **startsrc**, **stopsrc**

Scripts: **topsvcsctrl**

---

topsvcsctrl Command

Purpose

Starts the topology services subsystem.

Syntax

topsvcsctrl { **-a** | **-s** | **-k** | **-d** | **-c** | **-u** | **-t** | **-o** | **-r** | **-h** }

Description

The **topsvcsctrl** control script controls the operation of the topology services subsystem. The subsystem is under the control of the system resource controller (SRC) and belongs to a subsystem group called **topsvcs**. This script is normally started by the HACMP/ES startup process.

An instance of the topology services subsystem runs on every node of a cluster.

From an operational point of view, the topology services subsystem group is organized as follows:

Subsystem	topology services
Subsystem group	**topsvcs**
SRC subsystem	**topsvcs** The **topsvcs** subsystem is associated with the **hatsd** daemon and the **topsvcs** script. The **topsvcs** script configures and starts the **hatsd** daemon. The subsystem name on the nodes is **topsvcs**. There is one of each subsystem per node and it is associated with the cluster to which the node belongs.
Daemons	**hatsd** Provides the topology services. The **topsvcs** script configures and starts the **hatsd** daemon.

The **topsvcsctrl** script is not normally executed from the command line. It is normally called by the HACMP/ES startup command.

The **topsvcsctrl** script provides a variety of controls for operating the topology services subsystems:

- Adding, starting, stopping, and deleting the subsystems
- Cleaning up the subsystems, that is, deleting them from all system partitions
- Turning tracing on and off
- Refreshing the subsystem

Before performing any of these functions, the script obtains the current cluster name (using the **cllsclstr** command) and the node number (using the **clhandle** command). If the node number is **0**, the control script is running on the control workstation.

Except for the clean and unconfigure functions, all functions are performed within the scope of the current system partition.

Adding the subsystem: When the **-a** flag is specified, the control script uses the **mkssys** command to add the topology services subsystem to the SRC. The control script operates as follows:

1. It makes sure the **topsvcs** subsystem is stopped.
2. It removes the **topsvcs** subsystem from the SRC (in case it is still there).
3. It adds the **topsvcs** subsystem to the SRC.

Starting the subsystem: When the **-s** flag is specified, the control script uses the **startsrc** command to start the topology services subsystem, **topsvcs**.

Stopping the subsystem: When the **-k** flag is specified, the control script uses the **stopsrc** command to stop the topology services subsystem, **topsvcs**.

Deleting the subsystem: When the **-d** flag is specified, the control script uses the **rmssys** command to remove the topology services subsystem from the SRC. The control script operates as follows:

1. It makes sure that the **topsvcs** subsystem is stopped.
2. It removes the **topsvcs** subsystem from the SRC using the **rmssys** command.
3. It removes the port number from the **/etc/services** file.

Cleaning up the subsystems: When the **-c** flag is specified, the control script stops and removes the topology services subsystems for all clusters partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the clusters, using the **stopsrc -g topsvcs** command.
2. It removes all entries for the **topsvcs** subsystem from the **/etc/services** file.

Turning tracing on: When the **-t** flag is specified, the control script turns tracing on for the **hatsd** daemon, using the **traceson** command.

Turning tracing off: When the **-o** flag is specified, the control script turns tracing off (returns it to its default level) for the **hatsd** daemon, using the **tracesoff** command.

Refreshing the subsystem: When the **-r** flag is specified, the control script refreshes the subsystem, using the **topsvcs refresh** command and the **refresh** command. It rebuilds the information about the node and adapter configuration in the global object data manager (ODM) and signals the daemon to read the rebuilt information.

Logging: While it is running, the topology services daemon (**hatsd**) provides information about its operation and errors by writing entries in a log file called **/var/ha/log/topsvcs.***cluster_name*.

Flags

-a	Adds the subsystem.
-s	Starts the subsystem.
-k	Stops the subsystem.
-d	Deletes the subsystem.
-c	Cleans the subsystems.
-u	Removes the topology services subsystem from all partitions.
-t	Turns tracing on for the subsystem.
-o	Turns tracing off for the subsystem.
-r	Refreshes the subsystem.
-h	Writes the script's usage statement to standard output.

Security

You must be running with an effective user ID of **root** to use this script.

Exit Status

0	Indicates that the script completed successfully.
1	Indicates that an error occurred.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX-domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to **/var/ha/soc/hats/server_socket.***partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in an HACMP environment only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

This script writes error messages (as necessary) to standard error.

Examples

1. To add the topology services subsystem to the SRC, enter:
   ```
   topsvcsctrl -a
   ```
2. To start the topology services subsystem, enter:
   ```
   topsvcsctrl -s
   ```
3. To stop the topology services subsystem, enter:
   ```
   topsvcsctrl -k
   ```
4. To delete the topology services subsystem from the SRC, enter:
   ```
   topsvcsctrl -d
   ```
5. To clean up the topology services subsystem, enter:
   ```
   topsvcsctrl -c
   ```
6. To turn tracing on for the topology services daemon, enter:
   ```
   topsvcsctrl -t
   ```
7. To turn tracing off for the topology services daemon, enter:
   ```
   topsvcsctrl -o
   ```

Location

/usr/sbin/rsct/bin/topsvcsctrl Contains the **topsvcsctrl** script

Files

/var/ha/log/topsvcs.*cluster_name*
Contains the log of the **hatsd** daemon on the cluster named **cluster_name**

Related Information

Commands: **lssrc**, **startsrc**, **stopsrc**

Scripts: **topsvcs**

---

touch Command

Purpose

Updates the access and modification times of a file.

Syntax

touch [**-a**] [**-c**] [**-m**] [**-f**] [**-r** *RefFile*] [*Time* | **-t** *Time*] { *File* ... | *Directory* ... }

Description

The **touch** command updates the access and modification times of each file specified by the *File* parameter of each directory specified by the *Directory* parameter. If you do not specify a value for the *Time* variable, the **touch** command uses the current time. If you specify a file that does not exist, the **touch** command creates the file unless you specify the **-c** flag.

The return code from the **touch** command is the number of files for which the times could not be successfully modified (including files that did not exist and were not created).

Note: Any dates beyond and including the year 2038 are invalid.

Flags

-a Changes the access time of the file specified by the *File* variable. Does not change the modification time unless **-m** is also specified.

-c Does not create the file if it does not already exist. No diagnostic messages are written concerning this condition.

-f Attempts to force the touch in spite of read and write permissions on a file.

-m Changes the modification time of *File*. Does not change the access time unless **-a** is also specified.

-r *RefFile* Uses the corresponding time of the file specified by the *RefFile* variable instead of the current time.

Time Specifies the date and time of the new timestamp in the format *MMDDhhmm*[YY], where:

- *MM* Specifies the month of the year (01 to 12).
- *DD* Specifies the day of the month (01 to 31).
- *hh* Specifies the hour of the day (00 to 23).
- *mm* Specifies the minute of the hour (00 to 59).
- *YY* Specifies the last two digits of the year. If the *YY* variable is not specified, the default value is the current year.

-t *Time* Uses the specified time instead of the current time. The *Time* variable is specified in the decimal form [[*CC*]*YY*]*MMDDhhmm*[**.***SS*] where:

CC Specifies the first two digits of the year.

YY Specifies the last two digits of the year.

MM Specifies the month of the year (01 to 12).

DD Specifies the day of the month (01 to 31).

hh Specifies the hour of the day (00 to 23).

mm Specifies the minute of the hour (00 to 59).

SS Specifies the second of the minute (00 to 59).

Notes:

1. The **touch** command calls the **utime ()** subroutine to change the modification and access times of the file touched. This may cause the **touch** command to fail when flags are used if you do not actually own the file, even though you may have write permission to the file.
2. Do not specify the full path name **/usr/bin/touch** if you receive an error message when using the **touch** command.

Exit Status

This command returns the following exit values:

0 The command executed successfully. All requested changes were made.
>0 An error occurred.

Examples

1. To update the access and modification times of a file, enter:

   ```
   touch program.c
   ```

   This sets the last access and modification times of the `program.c` file to the current date and time. If the `program.c` file does not exist, the **touch** command creates an empty file with that name.
2. To avoid creating a new file, enter:

   ```
   touch  -c program.c
   ```
3. To update only the modification time, enter:

   ```
   touch  -m *.o
   ```

   This updates the last modification times (not the access times) of the files that end with a `.o` extension in the current directory. The **touch** command is often used in this way to alter the results of the **make** command.
4. To explicitly set the access and modification times, enter:

   ```
   touch  -c  -t 02171425 program.c
   ```

   This sets the access and modification dates to 14:25 (2:25 p.m.) February 17 of the current year.
5. To use the time stamp of another file instead of the current time, enter:

   ```
   touch  -r file1 program.c
   ```

   This gives the `program.c` file the same time stamp as the `file1` file.

6. To touch a file using a specified time other than the current time, enter:

```
touch  -t 198503030303.55 program.c
```

   This gives the `program.c` file a time stamp of 3:03:55 a.m. on March 3, 1985.

Files

/usr/bin/touch	Contains the **touch** command.

Related Information

The **date** command, **locale** command.

The **utime** subroutine.

Directories in *Operating system and device management* describes the structure and characteristics of directories in the file system.

Files in *Operating system and device management* describes files, file types, and how to name files.

File and directory access modes in *Operating system and device management* introduces file ownership and permissions to access files and directories.

Understanding File Types in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs* introduces the commands that control files.

---

tprof Command

Purpose

Reports CPU usage.

Syntax

tprof [**-c**] [**-C** { **all** | *CPUList* }] [**-d**] **-D**] [**-e**] [[**-E** [*mode*]] [**-f** *frequency*]] [**-F**] [**-l**] [**-j**] [**-k**] [**-l**] [**-L** *ObjectList*] [**-m** *ObjectsList*] [**-M** *SourcePathList*] [**-N**] [**-p** *ProcessList*] [**-P** { **all** | *PIDsList* }] [**-s**] [**-S** *SearchPathList*] [**-t**] [**-T** *BufferSize*] [**-u**] [**-v**] [**-V** *VerboseFileName*] { { **-r** *RootString* } | { [**-A** { **all** | *CPUList* }] [**-r** *RootString*] **-x** *Program* } }

Notes:

1. All the list type inputs are separated by a comma except for pathlist, which is separated by a colon.
2. Per-CPU profiling mode is automatically disabled while running in realtime mode.
3. Microprofiling is automatically disabled if per-CPU profiling is turned on.
4. If the **-x** flag is specified without the **-A** flag, **tprof** runs in realtime mode.
5. If the **-x** flag is specified with the **-A** flag, **tprof** runs in automated offline mode.
6. If the **-x** flag is omitted **tprof** runs in post-processing mode or manual offline mode, depending on the presence of cooked files and the **-F** flag.

Description

The **tprof** command reports CPU usage for individual programs and the system as a whole. This command is a useful tool for anyone with a JAVA, C, C++, or FORTRAN program that might be CPU-bound and who wants to know which sections of the program are most heavily using the CPU.

The **tprof** command can charge CPU time to object files, processes, threads, subroutines (user mode, kernel mode and shared library) and even to source lines of programs or individual instructions. Charging CPU time to subroutines is called profiling and charging CPU time to source program lines is called micro-profiling.

For subroutine-level profiling, the **tprof** command can be run without modifying executable programs, that is no recompilation with special compiler flags is necessary. This is still true if the executables have been stripped, unless the traceback tables have also been removed. However, recompilation is required to get a micro-profile, unless a listing file is already available. To perform micro-profiling on a program, either the program should be compiled with **-g** and the source files should be accessible to **tprof** or the program should be compiled with **-q***list* and either both the object listing files and the source files or just the object listing files should be accessible to **tprof**. To take full advantage of **tprof** micro-profiling capabilities, it is best to provide both the `.lst` and the source file.

The **tprof** command can run in the following modes:

1. Realtime or online
2. Manual offline
3. Automated offline
4. Post-processing

If the **-x** flag is specified but the **-A** flag is not specified, **tprof** runs in realtime mode. In realtime mode **tprof** starts the AIX **trace** utility in the background, and processes the trace data as it gets generated. When the program being profiled ends, **tprof** collects symbolic name information, and generates the **tprof** reports.

Note: This mode does not allow per-CPU profiling.

If the **-A** flag is specified with the **-x** flag, **tprof** runs in automated offline mode. In this mode **tprof** starts the AIX **trace** utility and logs the trace data into a file. Once the trace data collection is done, it collect symbolic name information, and **tprof** opens the trace log file and processes the data to generate reports. In this mode **tprof** generates the following files in addition to tprof report files:

1. *RootString*.syms
2. *RootString*.trc[-cpuid]

Note: All of the input and report files used by tprof are named *RootString.suffix*, where *RootString* is either specified with **-r** or is the program name specified with **-x**.

Note: In realtime mode and automated offline mode, the ulimit value of the data area for the program that is being profiled is set to **unlimited**.

Note: In automated offline mode, **-N** flag is specified to collect source line info into the generated **RootString.syms** file. And **-I** flag is specified to collect binary instructions into the generated **RootString.syms** file.

The **tprof** command can re-process these files any time to generate profiling reports. This is called manual offline mode. The *RootString***.syms** file contains symbolic name information similar to the output of the **gensyms** command. The *RootString.trc***[-cpuid**] file(s) are trace log file(s). The **-cpuid** is added to the names when per-CPU tracing is on. In that case, each file contains trace data from one CPU only.

Note: If **-c** is used along with **-A** then those two files are not generated. Instead, the following two files are created:

1. *RootString***.csyms**
2. *RootString***.ctrc**[**-cpuid**]

Those files are *cooked*, that is they are a pre-processed version of the normal trace and name files. **tprof** post-processes cooked file much faster.

If neither **-A** flag nor **-x** are specified, **tprof** runs either in manual offline or in post-processing mode. For **tprof** to run in post-processing mode the following files must be available:

1. *RootString***.csyms**
2. *RootString***.ctrc**[**-cpuid**]

Those are generated when **tprof** runs (in any mode except post-processing mode) with the **-c** flag. For **tprof** to run in manual offline mode the following files must be available:

1. *RootString***.syms**
2. *RootString***.trc** [**-cpuid**]

These files are generated by a user manually using the **gensyms** command and AIX trace facility or by running **tprof** in automated offline mode without the **-c** flag.

Note: **tprof** always first looks for *RootString.csyms* and *RootString***.ctrc**[**-cpuid**]. Only if these files are not available, does it look for *RootString***.syms** and *RootString***.trc**[**-cpuid**]. To prevent **tprof** from looking for *rootstring***.csyms** and *RootString***.ctrc**[**-cpuid**], that is, force the manual offline mode, use the **-F** flag.

Note: In automated offline mode, **-N** flag is specified to collect source line info into the generated **RootString.syms** file. And **-I** flag is specified to collect binary instructions into the generated **RootString.syms** file.

tprof generates a **tprof** report file named *RootString***.prof** which holds the process, thread, object file and subroutine level profiling report. The file can contains the following sections and subsections:

1. Summary report section
   a. CPU usage summary by process name
   b. CPU usage summary by threads (tid)
2. Global (pertains to the execution of all processes on system) profile section
   a. CPU usage of user mode routines
   b. CPU usage of kernel routines, including milicode routines called in kernel mode
   c. CPU usage summary for kernel extensions
   d. CPU usage of each kernel extension's subroutines
   e. CPU usage summary for shared libraries, including milicode routines called in user mode
   f. CPU usage of each shared library's subroutines
   g. CPU usage of each JAVA class
   h. CPU usage of each JAVA methods of each JAVA class
3. Process and thread level profile sections (one section for each process or thread)
   a. CPU usage of user mode routines for this process/thread
   b. CPU usage of kernel routines for this process/thread, including milicode routines called in kernel mode
   c. CPU usage summary for kernel extensions for this process/thread
   d. CPU usage of each kernel extension's subroutines for this process/thread
   e. CPU usage summary for shared libraries for this process/thread, including milicode routines called in user mode
   f. CPU usage of each shared library's subroutines for this process/thread
   g. CPU usage of each JAVA class for this process/thread
   h. CPU usage of JAVA methods of each JAVA class for this process/thread

The summary report section is always present in the *RootString***.prof** report file. Based on the profiling flags the various subsections of the global profile section can be turned on and off:

- **-u** turns on subsections a

- **-k** turns on subsection b
- **-e** turns on subsections c and d
- **-s** turns on subsections e and f
- **-j** turns on subsections g and h

The process and thread level profile sections are created for processes and threads selected with the **-p**, **-P** and **-t** flags. The subsections present within each of the per-process of per-thread sections are identical to the subsections present in the global section, they are selected using the profiling flags (**-u**,**-s**,**-k**,**-e**,**-j**).

Optionally (when called with the **-C** flag) **tprof** also generates per-CPU profiling reports (one profiling report per CPU). The generated **tprof** reports have the same structure and are named using the convention: *RootString*.**prof**[**-cpuid**].

If **-m** is specified, **tprof** generates micro-profiling reports. The reports use the following naming convention: *RootString.source*.**mprof**, where source is the base name of a source file. If more than one source file has the same base name, then a number to uniquely identify them is appended to the report file name, for example, *RootString.Filename*.**c.mprof-1**. The micro-profiling report has the following information:

1. The full path name of the annotated source file.
2. A hot line profile section which has all the line numbers from that source file hit by profiling samples, sorted by cpu usage. For each source line, one line reports the percentage of time spent on behalf of all processes, followed by additional lines with the breakdown by individual process.
3. A source line profile section for each of the functions in that source file, which have CPU usage. This section contains the source line number, cpu usage and source code. If a .lst file for that source file is accessible to tprof then it will interlace the instruction lines from the .lst file with the source lines from the source file and charge CPU usage appropriately. This will provide breakdown by instruction for each source file.

   If a source file is not present, but a **.lst** file is present, **tprof** only shows the CPU usage based on the source lines and the instructions from the .lst file.

   If neither files are present, but the source file was compiled with **-g**, **tprof** can retrieve the source line numbers and it generates a similar report, but with the source code column missing.

   Note: If per-CPU profiling is requested then micro-profiling is automatically disabled. **tprof** cannot report correct source line information if a **.c** file is included in another **.c** file. **tprof** cannot micro-profile JAVA classes or methods.

   Note: If **-m** is specified, **-N** is automatically specified to gather the source line info into a symbols file in automated offline mode.

If the **-Z** flag is used with the **-m** flag, there will be one report file generated per subroutine. The following naming convention is then used: **RootString.source.routine.mprof**, where *routine* is the name of one of the subroutines listed in the source file. In addition, a file named **RootString.source.HOT_LINES.mprof** containing the hot line profiling information described above is also created.

If the **-L** flag is specified, the **tprof** command generates annotated listing files. The files use the following naming convention: **RootString.source.alst**, where *source* is the base name of a source file. If more than one source file has the same base name, then a number to uniquely identify them is appended to the report file name. For example, **RootString.Filename.c.alst-1**. If the **-Z** flag is used with the **-L** flag, there will be one report file generated per subroutine. The following naming convention is then used: **RootString.source.routine.alst**, where *routine* is the name of one of the subroutines listed in the source file.

If **-N** flag or **-I** flag is specified when profiling a java program using JPA (**-x java -Xrunjpa**) , the JIT source line number and instructions can be collected if the corresponding parameter is added to the **-Xrunjpa** flag:

- **source=1** turns on JIT source line collecting (requires IBM JRE 1.5.0).
- **instructions=1** turns on JIT instructions collecting.

The following restrictions apply for non-root users running the **tprof** command:

1. **tprof** will not be able to verify that the running kernel is the same as the **/unix** file. This means that even if a warning message is displayed, in most cases the running kernel and **/unix** are the same, so the data should be accurate.
2. When the **gensyms** command is run by a non-root user, the same warning as in restriction #1 (above) is given and the **gensyms** file is marked. If **tprof** is run in the offline mode, the file created with the **gensyms** command will flag **tprof** as to unVerified kernel.
3. **tprof** will not be able to open and read symbols on files which do not have the read permission set. Some private, shared libraries do not have read permission, and some kernel extensions are nonReadable.

Time-Based versus Event-Based Profiling

By default, **tprof** is time-based and is driven by the decrementer interrupt. Another mode of profiling is event-based profiling, in which the interrupt is driven by either software-based events or by Performance Monitor events. With event-based profiling, both the sampling frequency and the profiling event can be varied on the command line.

The **-E** flag enables event-based profiling. The **-E** flag parameter is one of the four software-based events (EMULATION, ALIGNMENT, ISLBMISS, DSLBMISS) or a Performance Monitor event (PM_*). By default, the profiling event is processor cycles. All Performance Monitor events are prefixed with PM_, such as PM_CYC for processor cycles or PM_INST_CMPL for instructions completed. **pmlist** lists all Performance Monitor events that are supported on a processor.

The **-f** flag varies the sampling frequency for event-based profiling. For software-based events and processor cycles, supported frequencies range from 1 to 500 milliseconds, with a default of 10 milliseconds. For all other Performance Monitor events, the range is from 10,000 to MAXINT occurrences of the event, with a default of 10,000 events.

Additional information is added to the **.prof** file to reflect the processor name, profiling event, and sampling frequency.

Java™ Applications Profiling

To profile JAVA applications, the **-j** flag must be specified, and the applications must be started with the **-Xrunjpa java** command line option. When this option is selected, the JVM will automatically make calls to the **jpa** library whenever new classes and methods are loaded into memory. The library will in turn collect address to name mapping information for methods and classes in files named **/tmp/Java***PID***.syms**, where *PID* is the process ID of a process running a JAVA virtual machine. The **tprof** command will automatically look in that directory for such files.

When running in automated offline mode, or selecting the cooking option, the **tprof** command will copy the information contained in **JavaPID.syms** files into the **RootString.syms** or **RootString.csyms** file. The corresponding files in **/tmp** can then be deleted. The directory content should be kept up to date by **tprof** command users. Whenever the JVM corresponding to a particular **JavaPID.syms** is stopped, the file should be deleted.

Profile Accuracy

The degree to which CPU activity can be resolved is determined by the number of samples captured and the degree to which *hot spots* dominate. While a program with a few hot spots can be profiled with relatively few samples, less-frequently executed sections of the program are not visible in the profiling reports unless more samples are captured. In cases where user programs run less than a minute, there may be insufficient resolution to have a high degree of confidence in the estimates.

A simple solution is to repeatedly execute the user program or script until you achieve the degree of resolution you need. The longer a program is run, the finer the degree of resolution of the profile. If you doubt the accuracy of a profile, run the **tprof** command several times and compare the resulting profiles.

Comparison of tprof Versus prof and gprof

The most significant differences between these three commands is that **tprof** collects data with no impact on the execution time of the programs being profiled, and works on optimized and stripped binaries without any need for recompilation, except to generate micro-profiling reports. Neither **gprof** nor **prof** have micro-profiling capabilities or work on optimized binaries, while they do require special compilation flags, and induce a slowdown in the execution time that can be significant. **prof** does not work on stripped binaries.

The **prof** and **gprof** tools are standard, supported profiling tools on many UNIX systems, including this operating system. Both **prof** and **gprof** provide subprogram profiling and exact counts of the number of times every subprogram is called. The **gprof** command also provides a very useful *call graph* showing the number of times each subprogram was called by a specific parent and the number of times each subprogram called a child. The **tprof** command provides neither subprogram call counts nor call graph information.

Like the **tprof** command, both the **prof** and **gprof** commands obtain their CPU consumption estimates for each subprogram by sampling the program counter of the user program.

tprof collects CPU usage information for the whole system, while **prof** and **gprof** collect only profiling information for a single program and only for the time spent in user mode.**tprof** also provides summary for all processes active during the execution of the profiled user program and fully support libraries and kernel mode profiling.

tprof support the profiling of JAVA applications, which **prof** and **gprof** do not.

Flags

-A {**all** | *CPUList* }
: Turns on automatic offline mode. No argument turns off per-CPU tracing. **all** enables tracing of all CPUs. *CPUList* is a comma separated list of CPU-ids to be traced.

-c
: Turns on generation of cooked files.

-C all | *CPUList*
: Turns on the per-CPU profiling. Specify **all** to generate profile reports for all CPUs. CPU numbers should be separated with a comma if you give a *CPUlist* (for example, 0,1,2). **Note:** per-CPU profiling is possible only if per-CPU trace is either on (in automated offline mode), or has been used (in manual offline mode). It is not possible at all in online mode.

-d
: Turns on deferred tracing mode, that is defers data collection until **trcon** is called.

-D
: Turns on detailed profiling which displays CPU usage by instruction offset under each subroutine.

-e
: Turns on kernel extension profiling.

-E [*mode*]
: Enables event-based profiling. The possible modes are:

  PM_*event*
  : Specifies the hardware event to profile. If no mode is specified for the **-E** flag, the default event is processor cycles (**PM_CYC**).

  EMULATION
  : Enables the emulation profiling mode.

  ALIGNMENT
  : Enables the alignment profiling mode.

  ISLBMISS
  : Enables the Instruction Segment Lookaside Buffer miss profiling mode.

  DSLBMISS
  : Enables the Data Segment Lookaside Buffer miss profiling mode.

-f *frequency*
Specifies the sampling frequency. The sampling frequency can be from 1 to 500 milliseconds for processor cycles and **EMULATION**, **ALIGNMENT**, **ISLBMISS**, and **DSLBMISS** events, and from 10000 to **MAXINT** event occurences for other Performance Monitor events.

-F
Overwrites cooked files if they exists. If used without the **-x** flag, this forces the manual offline mode.

-I
Turn on binary instructions collecting.
Note: **-I** activates to gather binary instructions when generating symbol files or cooked symbol files in automated offline mode. However, in maunal offline mode, **-I** does not affect the report files.

-j
Turns on JAVA classes and methods profiling.

-k
Enables kernel profiling.

-l
Enables long names reporting. By default **tprof** truncates the subroutine, program and source file names if they do not fit into the available space in the profiling report. This flag disables truncation.

-L *ObjectList*
Enables listing annotation for objects specified by the comma separated list, *Objectlist*. Executables and shared libraries can have their listing files annotated. Specify the archive name for libraries.

Notes:

1. To enable listing annotation of programs, user mode profiling (**-u**) must be turned on.
2. To enable listing annotation of shared libraries, shared library profiling (**-s**) must be turned on.
3. To annotate a listing generated with IPA compilations, specify `a.lst` as the *ObjectList*.

-m *ObjectsList*
Enables micro-profiling of objects specified by the comma separated list, *Objectlist*. Executables, shared libraries, and kernel extensions can be micro-profiled. Specify the archive name for libraries and kernel extensions.

Notes:

1. To enable micro-profiling of programs, user mode profiling (**-u**) must be turned on.
2. To enable micro-profiling of shared libraries, shared library profiling (**-s**) must be turned on.
3. To enable micro-profiling of kernel extensions, kernel extension profiling (**-e**) must be turned on.

-M *PathList*
Specifies the source path list. The *PathList* is a colon separated list of paths that are searched for source files and **.lst** files that are required for micro-profiling and listing annotation.
Note: By default the source path list is the object search path list.

-N
Turn on source line number info collecting.
Note: **-N** activates to gather source line information when generating symbol files or cooked symbol files in automated offline mode. However, in maunal offline mode, **-N** does not affect the report files.

-p *ProcessList*
Enables process level profiling of the process names specified in the *ProcessList*. *ProcessList* is a comma separated list of process names
Note: Process level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, or **-j**) is turned on.

-P { **all** | *PIDList* }
Enables process level profiling of **all** processes encountered or for processes specified with *PIDList*. The *PIDList* is a comma separated list of process-IDs.
Note: Process level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, or **-j**) is turned on.

-r *RootString*
Specifies the *RootString*. **tprof** input and report files all have names in the form of *RootString.suffix*.
Note: If **-r** is not specified, *RootString* defaults to the program name specified with the **-x** flag.

-R	Specifies that the **tprof** command should use samples weighted by PURR increment values to calculate percentages. This is the preferred mode when running in either simultaneous multi-threading or Micro-Partitioning environments. **Note:** The **-R** flag cannot be used with either the **-z** flag or the **-Z** flag.
-s	Enables shared library profiling.
-S *PathList*	Specifies the object search *PathList*. The *PathList* is a colon separated list of paths that are searched for executables, shared libraries and kernel extensions. **Note:** The default object search *PathList* is the environment path list (**$PATH**).
-t	Enables thread level profiling. **Note:** If **-p** or **-P** are not specified with the **-t** flag, **-t** is equivalent to **-P all -t**. Otherwise, it enables thread level reporting for the selected processes. Thread level profiling is enabled only if at least one of the profiling modes (**-u**,**-s**,**-k**,**-e**, **-j**) is enabled..
-T *BufferSize*	Specifies the trace *BufferSize*. **Note:** This flag has meaning only in real time or automated offline modes.
-u	Enables user mode profiling.
-v	Enables verbose mode.
-V *File*	Stores the verbose output in the specified *File*.
-x *Program*	Specifies the program to be executed by **tprof**. Data collection stops when *Program* completes or **trace** is manually stopped with either **trcoff** or **trcstop** **Note:** The **-x** flag must be the last flag in the list of flags specified in **tprof**.
-z	Enables compatibility mode with the previous version of **tprof**. By default CPU usage is only reported in percentages. When **-z** is used, **tprof** also reports ticks. This flag also adds the Address and Bytes columns in subroutine reports.
-Z	Switches reports to use ticks instead of percentages (same as the **-z** flag), and splits annotated listing (when used with the **-L** flag) and annotated source files (when used with the **-m** flag) into multiple files, one per subroutine.

Examples

1. Basic global program and thread-level summary

   ```
   $ tprof -x sleep 10
   Mon May 20 00:39:26 2002 System: AIX 5.2 Node: dreaming Machine: 000671894C00
   Starting Command sleep 10
   stopping trace collection.
   Generating sleep.prof
   ```

   The resulting **sleep.prof** file only contains the summary report section.

2. Global profiling with all options

   ```
   $ tprof -skeuj -x sleep 10
   Mon May 20 00:39:26 2002
   System: AIX 5.2 Node: drea
   ming Machine: 000671894C00
   Starting Command sleep 10
   stopping trace collection.
   Generating sleep.prof
   ```

   The resulting **sleep.prof** file contains the summary report and global profile sections.

3. Single process level profiling

   ```
   $ tprof -u -p workload -x workload
   Mon May 20 00:39:26 2002
   System: AIX 5.2 Node: drea
   ming Machine: 000671894C00
   Starting Command workload stopping trace collection.
   Generating workload.prof
   ```

   The resulting **workload.prof** file contains the summary report and the global user mode profile sections, and one process level profile section for the process 'workload' that contains only a user mode profile subsection.

4. Multiple process level profiling

```
$ tprof -se -p send,receive -x startall
Mon May 20 00:39:26 2002
System: AIX 5.2 Node: dreaming Machine: 000671894C00
Starting Command startall
stopping trace collection.
Generating startall.prof
```

The resulting **startall.prof** file contains the summary report, the global shared library mode profile, and the global kernel extension profile sections, and two process level profile sections: one for the process 'send', and one for the process 'receive'. The process level sections each contain two subsections: one with shared library profiling information and one with kernel extensions profiling information.

5. Microprofiling and listing annotation

```
$ tprof -m ./tcalc -L ./tcalc -u -x ./tcalc
Mon May 20 00:47:09 2002
System: AIX 5.2 Node: dreaming Machine: 000671894C00
Starting Command ./tcalc
stopping trace collection.
Generating tcalc.prof
Generating tcalc.tcalc.c.mprof
Generating tcalc.tcalc.c.alst
```

The resulting **tcalc.prof** file contains the summary report and the global user mode profile sections. The resulting **tcalc.tcalc.c.mprof** and **tcalc.tcalc.c.alst** files contain the micro-profiling report and the annotated listing.

6. For event-based profiling on processor cycles, sampling once every 100 milliseconds, type:

```
tprof -E -f 100 -Askex sleep 10
```

Output similar to the following displays:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:44:02 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

7. For event-based profiling on completed instructions, sampling once every 20,000 completed instructions, type:

```
tprof -E PM_INST_CMPL -f 20000 -Askex sleep 10
```

Output similar to the following displays:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:42:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

8. For event-based profiling on emulation interrupts, sampling once every 10,000 events, type:

```
tprof -E EMULATION -Askex sleep 10
```

Output similar to the following displays:

```
Starting Command sleep 10
stopping trace collection.
Tue Apr 26 14:41:44 2005
System: AIX 5.3 Node: bigdomino Machine: 00C0046A4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

9. Automated offline mode

```
tprof -c -A all -x sleep 10
Starting Command sleep 10
stopping trace collection.
```

```
Mon May 20 00:52:52 2002
System: AIX 5.2 Node: dreaming Machine: 000671894C00
Generating sleep.ctrc
Generating sleep.csyms
Generating sleep.prof
```

The resulting **sleep.prof** file only has a summary report section, while the two cooked files are ready to be re-postprocessed.

10. Automated offline mode enabling source line collecting.

```
$ tprof -A -N -x sleep 10
Starting Command sleep 10
stopping trace collection.
Wed Feb  8 15:12:41 2006
System: AIX 5.3 Node: aixperformance Machine: 000F9F3D4C00
Generating sleep.trc
Generating sleep.prof
Generating sleep.syms
```

The resulting **sleep.prof** file only contains the summary report section, while **sleep.syms** contains the source line information.

11. Automated offline mode enabling source line and instruction collecting.

```
$ tprof -A -N -I -r RootString -x sleep 10
Starting Command sleep 10
stopping trace collection.
Wed Feb  8 15:16:37 2006
System: AIX 5.3 Node: aixperformance Machine: 000F9F3D4C00
Generating RootString.trc
Generating RootString.prof
Generating RootString.syms
```

The resulting **rootstring.prof** file is generated, while **rootstring.syms** contains the source line information and binary instructions.

12. Realtime mode enabling java source line and instructions collecting.

```
$ tprof -N -I -x java -Xrunjpa:source=1,instructions=1 HelloAIX
Thu Feb  9 13:30:38 2006
System: AIX 5.3 Node: perftdev Machine: 00CEBB4A4C00
Starting Command java -Xrunvpn_jpa:source=1,instructions=1 HelloAIX
Hello AIX!
stopping trace collection.
Generating java.prof
```

The resulting **java.prof** file is generated, and it contains the JIT source line information as well as JIT instructions.

Messages

If your system displays the following message:

```
/dev/systrace: device busy or trcon: TRCON:no such device
```

This means the **trace** facility is already in use. Stop your program and try again after typing `trcstop`, stops the trace.

Related Information

The **gensyms** command, **gprof** command, **prof** command, **stripnm** command, and **trace** daemon.

tput Command

Purpose

Queries the **terminfo** database for terminal-dependent information.

Syntax

For Outputting Terminal Information

tput [**-T** *Type*] [*CapabilityName {clear, init, longname, reset}* [*Parameters*...]]

For Using stdin to Process Multiple Capabilities

tput [**-S**]

Description

The **tput** command uses the terminfo database to make terminal-dependent information available to the shell. The **tput** command outputs a string if the attribute *CapabilityName* is of type *string.* The output string is an integer if the attribute is of type *integer*. If the attribute is of type *Boolean*, the **tput** command sets the exit value (0 for TRUE, 1 for FALSE), and produces no other output.

XTERM DESCRIPTION LIMITATION

The xterm terminal description in the DEC.TI file on AIX Version 4 provides underline mode by using the SGR attribute. The SMUL and RMUL attributes are not currently defined in the XTERM terminal description on AIX Version 4. Use the more generic capability named SGR.

```
tput sgr x y
```

Where *x* is either a 1 or a 0 to turn standout mode on or off respectively, and *y* is either a 1 or a 0 to turn underline mode on or off respectively. See the article ″**terminfo** file format″ for more details on the SGR capability.

```
tput sgr 0 1     turn off standout; turn on underline
tput sgr 0 0     turn off standout; turn off underline
tput sgr 1 1     turn on standout; turn on underline
tput sgr 1 0     turn on standout; turn off underline
```

Flags

In addition to the capability names, the following strings are supported as arguments to the **tput** subroutine.

clear	Displays the clear screen sequence (this is also a capability name).
init	Displays the sequence that initializes the user's terminal in an implementation-dependent manner.
reset	Displays the sequence that will reset the user's terminal in an implementation-dependent manner.
longname	Displays the long name and the specified terminal (or current terminal if none specified).
-S	Uses stdin. This allow the tput to process multiple capabilities. When using the -S option, the capabilities cannot be entered on the command line. Enter ^D token finished.
-T*Type*	Indicates the type of terminal. If **-T** is not specified, the **TERM** environment variable is used for the terminal.

Exit Status

This command returns the following exit values:

0	The requested string was written successfully.
1	Unspecified.
2	Usage error.
3	No information is available about the specified terminal type.

4 The specified operand is invalid.
>4 An error occurred.

Examples

1. To clear the screen for the current terminal, enter:
   ```
   tput clear
   ```
2. To display the number of columns for the current terminals, enter:
   ```
   tput cols
   ```
3. To display the number of columns for the aixterm terminal, enter:
   ```
   tput  -Taixterm cols
   ```
4. To set the shell variable **bold** to the begin standout mode sequence and the shell variable **offbold** to the end standout mode sequence, enter:
   ```
   bold=`tput smso`

   offbold='tput rmso'
   ```
   Entering these commands might be followed by the following prompt:
   ```
   echo "${bold}Name: ${offbold} \c"
   ```
5. To set the exit value to indicate if the current terminal is a hardcopy terminal, enter:
   ```
   tput hc
   ```
6. To initialize the current terminal, enter:
   ```
   tput init
   ```

Files

/usr/share/lib/terminfo/?/*	Contains the terminal descriptor files.
/usr/include/term.h	Contains the definition files.

Related Information

The **stty** command.

The **terminfo** file format.

---

tr Command

Purpose

Translates characters.

Syntax

tr [**-c** | **-cds** | **-cs** | **-C** | **-Cds** | **-Cs** | **-ds** | **-s**] [**-A**] *String1 String2*

tr { **-cd** | **-cs** | **-Cd** | **-Cs** | **-d** | **-s** } [**-A**] *String1*

Description

The **tr** command deletes or substitutes characters from standard input and writes the result to standard output. The **tr** command performs three kinds of operations depending on the strings specified by the *String1* and *String2* variable and on the flags specified.

Transforming Characters

If *String1* and *String2* are both specified and the **-d** flag is not specified, the **tr** command replaces each character contained in *String1* from the standard input with the character in the same position in *String2*.

Deleting Characters Using the -d Flag

If the **-d** flag is specified, the **tr** command deletes each character contained in *String1* from standard input.

Removing Sequences Using the -s Flag

If the **-s** flag is specified, the **tr** command removes all but the first character in any sequence of a character string represented in *String1* or *String2*. For each character represented in *String1*, the **tr** command removes all but the first occurrence of the character from standard output. For each character represented in *String2*, the **tr** command removes all but the first occurrence in a sequence of occurrences of that character in the standard output.

Special Sequences for Expressing Strings

The strings contained in the *String1* and *String2* variables can be expressed using the following conventions:

C1-*C2*
: Specifies the string of characters that collate between the character specified by *C1* and the character specified by *C2*, inclusive. The character specified by *C1* must collate before the character specified by *C2*.
Note: The current locale has a significant effect on results when specifying subranges using this method. If the command is required to give consistent results irrespective of locale, the use of subranges should be avoided.

[*C*******Number***]**
: *Number* is an integer that specifies the number of repetitions of the character specified by *C*. *Number* is considered a decimal integer unless the first digit is a 0; then it is considered an octal integer.

[*C****]**
: Fills out the string with the character specified by *C*. This option, used only at the end of the string contained within *String2*, forces the string within *String2* to have the same number of characters as the string specified by the *String1* variable. Any characters specified after the ***** (asterisk) are ignored.

[:*ClassName***:]**
: Specifies all of the characters in the character class named by *ClassName* in the current locale. The class name can be any of the following names:

```
alnum      lower
alpha      print
blank      punct
cntrl      space
digit      upper
graph      xdigit
```

Except for **[:**`lower`**:]** and **[:**`upper`**:]** conversion character classes, the characters specified by other character classes are placed in the array in an unspecified order. Because the order of the characters specified by character classes is undefined, the characters should be used only if the intent is to map several characters into one. An exception to this is the case of conversion character classes.

For more information on character classes, see the **ctype** subroutines.

[=*C***=]**
: Specifies all of the characters with the same equivalence class as the character specified by *C.*

*****Octal
: Specifies the character whose encoding is represented by the octal value specified by *Octal*. *Octal* can be a one-, two- or three-digit octal integer. The NULL character can be expressed with `'\0'`, and is processed like any other character.

*****ControlCharacter* Specifies the control character that corresponds to the value specified by *ControlCharacter*. The following values can be represented:

\a	Alert
\b	Backspace
\f	Form-feed
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab

**** Specifies the \ (backslash) as itself, without any special meaning as an escape character.

\[Specifies the [(left bracket) as itself, without any special meaning as the beginning of a special string sequence.

\- Specifies the - (minus sign) as itself, without any special meaning as a range separator.

If a character is specified more than once in *String1*, the character is translated into the character in *String2* that corresponds to the last occurrence of the character in *String1*.

If the strings specified by *String1* and *String2* are not the same length, the **tr** command ignores the extra characters in the longer string.

Flags

-A Performs all operations on a byte-by-byte basis using the ASCII collation order for ranges and character classes, instead of the collation order for the current locale.

-C Specifies that the value of *String1* be replaced by the *complement* of the string specified by *String1*. The complement of *String1* is all of the characters in the character set of the current locale, *except* the characters specified by *String1*. If the **-A** and **-c** flags are both specified, characters are complemented with respect to the set of all 8-bit character codes. If the **-c** and **-s** flags are both specified, the **-s** flag applies to characters in the complement of *String1*.

If the **-d** option is not specified, the complements of the characters specified by *String1* will be placed in the array in ascending collation sequence as defined by the current setting of **LC_COLLATE**.

-c Specifies that the value of *String1* be replaced by the *complement* of the string specified by *String1*. The complement of *String1* is all of the characters in the character set of the current locale, *except* the characters specified by *String1*. If the **-A** and **-c** flags are both specified, characters are complemented with respect to the set of all 8-bit character codes. If the **-c** and **-s** flags are both specified, the **-s** flag applies to characters in the complement of *String1*.

If the **-d** option is not specified, the complement of the values specified by *String1* will be placed in the array in ascending order by binary value.

-d Deletes each character from standard input that is contained in the string specified by *String1*.

Notes:

1. When the **-C** option is specified with the **-d** option, all characters except those specified by *String1* will be deleted. The contents of *String2* are ignored unless the **-s** option is also specified.
2. When the **-c** option is specified with the **-d** option, all values except those specified by *String1* will be deleted. The contents of *String2* are ignored unless the **-s** option is also specified.

-s Removes all but the first in a sequence of a repeated characters. Character sequences specified by *String1* are removed from standard input before translation, and character sequences specified by *String2* are removed from standard output.

String1 Specifies a string of characters.

String2 Specifies a string of characters.

Exit Status

This command returns the following exit values:

0 All input was processed successfully.
>0 An error occurred.

Examples

1. To translate braces into parentheses, type:

   ```
   tr '{}' '()' < textfile > newfile
   ```

   This translates each { (left brace) to ((left parenthesis) and each } (right brace) to) (right parenthesis). All other characters remain unchanged.
2. To translate braces into brackets, type:

   ```
   tr '{}' '\[]' < textfile > newfile
   ```

   This translates each { (left brace) to [(left bracket) and each } (right brace) to] (right bracket). The left bracket must be entered with a \ (backslash) escape character.
3. To translate lowercase characters to uppercase, type:

   ```
   tr 'a-z' 'A-Z' < textfile > newfile
   ```
4. To create a list of words in a file, type:

   ```
   tr -cs '[:lower:][:upper:]' '[\n*]' < textfile > newfile
   ```

   This translates each sequence of characters other than lowercase letters and uppercase letters into a single newline character. The * (asterisk) causes the **tr** command to repeat the new line character enough times to make the second string as long as the first string.
5. To delete all NULL characters from a file, type:

   ```
   tr -d '\0' < textfile > newfile
   ```
6. To replace every sequence of one or more new lines with a single new line, type:

   ```
   tr -s '\n' < textfile > newfile
   ```

   OR

   ```
   tr -s '\012' < textfile > newfile
   ```
7. To replace every nonprinting character, other than valid control characters, with a ? (question mark), type:

   ```
   tr -c '[:print:][:cntrl:]' '[?*]' < textfile > newfile
   ```

   This scans a file created in a different locale to find characters that are not printable characters in the current locale.
8. To replace every sequence of characters in the <space> character class with a single # character, type:

   ```
   tr -s '[:space:]' '[#*]'
   ```

Related Information

The **ed** command, **trbsd** command.

The **ctype** subroutines.

National Language Support Overview in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

trace Daemon

Purpose

Records selected system events.

Syntax

trace [**-a** [**-g**]] [**-f** | **-l**] [**-b** | **-B**] [**-c**] [**-C** [*CPUList* | **all**]] [**-d**] [**-e** *string-cmd*] [**-h**] [**-j** *Event* [*,Event*]] [**-k** *Event* [**,***Event*]] [**-J** *Event-group* [*,Event-group*]] [**-K** *Event-group* [*,Event-group*]] [**-m** *Message*] [-M] [**-n**] [**-o** *Name*] [**-o-**] [**-p**] [**-r** *reglist*] [**-s**] [**-A** *process-id* [*,process-id*]] [**-t** *thread-id* [*,thread-id*]] [**-x** *program-specification* | **-X** *program-specification*] [**-l**] [**-P** *trace-propagation*][**-L** *Size*] [**-T** *Size*]

Description

The **trace** daemon configures a trace session and starts the collection of system events. The data collected by the trace function is recorded in the trace log. A report from the trace log can be generated with the **trcrpt** command.

When invoked with the **-a**. **-x**, or **-X** flags, the trace daemon is run asynchronously (i.e. as a background task). Otherwise, it is run interactively and prompts you for subcommands.

You can use the System Management Interface Tool (SMIT) to run the **trace** daemon. To use SMIT, enter:

```
smit trace
```

The following are modes of trace data collection:

Alternate (the default)	All trace events are captured in the trace log file.
Circular (**-l**)	The trace events wrap within the in-memory buffers and are not captured in the trace log file until the trace data collection is stopped.
Single (**-f**)	The collection of trace events stops when the in-memory trace buffer fills up and the contents of the buffer are captured in the trace log file.
Buffer Allocation	Trace buffers are allocated from either the kernel heap, or are put into separate segments. By default, buffers are allocated from the kernel heap unless the buffer size requested is too large for buffers to fit in the kernel heap, in which case they are allocated in separate segments. Allocating buffers from separate segments hinders trace performance somewhat. However, buffers in separate segments will not take up paging space, just pinned memory. The type of buffer allocation can be specified with the optional **-b** or **-B** flags.

Beginning with AIX 5.3, you can elect to trace only selected processes or threads. You can also trace a single program. You can specify whether the trace is to be propagated or extended to newly created processes or threads. You can optionally include interrupt events in such traces. This is only valid for trace channel `0`.

Notes:

1. Unless the trace is started before the process that is being traced, the process startup events are not captured. If the trace is started before the process that is being traced, some events from processes other than the process being traced will be captured as well.
2. When trace uses memory from the kernel heap which is the case for the **-B** option (32-bit kernel only), this memory remains part of kernel memory until the next reboot of the system. Thus, care should be taken when using large buffers. Furthermore, it should be noted that, prior to AIX 5.2 with the 5200-05 Maintenance Level, when using the multi-CPU trace for all CPUs, the **-C all** option, memory is allocated for all possible CPUs, not just those running at the time the trace was started. This is fixed in AIX 5.2 with the 5200-05 Maintenance Level and AIX 5.3.

Flags

-a
Runs the **trace** daemon asynchronously (i.e. as a background task). Once **trace** has been started this way, you can use the **trcon**, **trcoff**, and **trcstop** commands to respectively start tracing, stop tracing, or exit the trace session. These commands are implemented as links to **trace.**

-A *process-id*[,*process-id*]
Traces only the listed processes and, optionally, their children. A *process-id* is a decimal number. Multiple process IDs can be separated by commas or enclosed in quotes and separated by spaces. The **-A** flag is only valid for trace channel `0`; the **-A** and **-g** flags are incompatible.

All threads existing for the specified processes when tracing is started are traced. By default, if after the trace starts, the processes being traced create additional threads or processes, these are not traced unless the **-P** flag is specified.

-b
Allocate buffers from the kernel heap. If the requested buffer space can not be obtained from the kernel heap, the command fails.
Note: The **-b** flag is only valid with the 32–bit kernel.

-B
Allocate buffers in separate segments.
Note: The **-B** flag is only valid with the 32–bit kernel.

-c
Saves the trace log file, adding **.old** to its name.

-C [*CPUList* | **all**]
Traces using one set of buffers per CPU in the *CPUList*. The CPUs can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. To trace all CPUs, specify **all**. Since this flag uses one set of buffers per CPU, and produces one file per CPU, it can consume large amounts of memory and file space, and should be used with care. The files produced are named **trcfile**, **trcfile-0**, **trcfile-1**, etc., where **0**, **1**, etc. are the CPU numbers. If **-T** or **-L** are specified, the sizes apply to each set of buffers and each file. On a uniprocessor system, you may specify `-C all`, but `-C` with a list of CPU numbers is ignored. Prior to AIX 5.3, if **-C** is used to specify more than one CPU, such as `-Call` or `-C "0 1"`, the associated buffers are not extracted from a system dump with **trcdead**.

Attention: The -C flag can only be used by the root user.

-d
Disables the automatic start of trace data collection. Delays starting of trace data collection. Normally, the collection of trace data starts automatically when you issue the **trace** daemon. Use the **trcon** command to start the collection of trace data.

-e *string-cmd*
Configures Component Trace by running **ctctrl** with *string-cmd* as an argument before the trace is started. In other words, it runs **ctctrl** *string-cmd*. Passing multiple **-e** options is allowed and is equivalent to successively running the **ctctrl** command with each *string-cmd* of arguments. This option can be used to configure the system trace mode (by setting the system trace mode to On, changing the level of trace, and so on) for some components just before starting to trace the system.

-f
Runs **trace** in a single mode. Causes the collection of trace data to stop as soon as the in-memory buffer is filled up. The trace data is then written to the trace log. Use the **trcon** command to restart trace data collection and capture another full buffer of data. If you issue the **trcoff** subcommand before the buffer is full, trace data collection is stopped and the current contents of the buffer are written to the trace log.

-g
Starts a trace session on a generic trace channel (channels 1 through 7). This flag works only when **trace** is run asynchronously (**-a**). The return code of the command is the channel number; the channel number must subsequently be used in the generic trace subroutine calls. To stop the generic trace session, use the command **trcstop -<**channel_number>.

-h
Omits the header record from the trace log. Normally, the **trace**daemon writes a header record with the date and time (from the **date** command) at the beginning of the trace log; the system name, version and release, the node identification, and the machine identification (from the **uname -a** command); and a user-defined message. At the beginning of the trace log, the information from the header record is included in the output of the **trcrpt** command.

-I
Trace interrupt events. When specified with **-A** or **-t**, the **-I** flag includes interrupt events along with the events for the processes or threads specified. When -I is specified, but neither **-A** nor **-t** is specified, only interrupt level events are traced. The **-I** flag is only valid for trace channel 0; the **-I** and **-g** flags are incompatible.

-j *Event*[**,***Event*]
-k *Event*[**,***Event*]
Specifies the user-defined events for which you want to collect (**-j**) or exclude (**-k**) trace data. The *Event* list items can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks.
Note: The following events are used to determine the pid, the cpuid and the exec path name in the **trcrpt** report:

```
106 DISPATCH
10C DISPATCH IDLE PROCESS
134 EXEC SYSTEM CALL
139 FORK SYSTEM CALL
465 KTHREAD CREATE
```

If any of these events is missing, the information reported by the **trcrpt** command will be incomplete. Consequently: when using the **-j** flag, you should include all these events in the *Event* list; conversely, when using the **-k** flag, you should not include these events in the *Event* list. If starting the trace with SMIT, or the -J flag, these events are in the **tidhk** group.

-J *Event-group* [**,** *Event-group*]
-K *Event-group* [**,***Event-group*]
Specifies the event groups to be included (**-J**) or excluded (**-K**). Event groups are described in ″Debug and Performance Tracing″. The **-J** and **-K** flags work like **-j** and **-k**, except with event groups instead of individual hook ids. All four flags, **-j**, **-J**, **-k**, and **-K** may be specified.

-l
Runs **trace** in a circular mode. The **trace** daemon writes the trace data to the trace log when the collection of trace data is stopped. Only the last buffer of trace data is captured. When you stop trace data collection using the **trcoff** command, restart it using the **trcon**command.

-L *Size*
Overrides the default trace log file size of 1MB with the value stated. Specifying a file size of zero sets the trace log file size to the default size. HERE

> **Note:** In the circular and the alternate modes, the trace log file size must be at least twice the size of the trace buffer. In the single mode, the trace log file must be at least the size of the buffer. See the **-T** flag for information on controlling the trace buffer size.

-m *Message*
Specifies text to be included in the message field of the trace log header record.

-M Dumps the address map of running processes into the trace. The **-M** flag must be specified if the trace file is to be processed by the **tprof** command.

-n Adds information to the trace log header: lock information, hardware information, and, for each loader entry, the symbol name, address, and type.

-o *Name* Overrides the **/var/adm/ras/trcfile** default trace log file and writes trace data to a user-defined file.

-o - Overrides the default trace log name and writes trace data to standard output. The **-c** flag is ignored when using this flag. An error is produced if **-o -** and **-C** are specified.

-p Includes the cpuid of the current processor with each hook. This flag is only valid for 64-bit kernel traces.

Note: The **trcrpt** command can report the cpuid whether or not this option is specified.

-P *propagation* The propagation is specified with the letters `p` for propagation across process creation, `t` for propagation across thread creation, and `n` for no propagation. Propagation across process creation implies propagation across thread creation. For example, if **-A** is specified to trace a process, all threads for that process that exist at the time the trace was started are traced. The **-Pt** flags causes all threads subsequently created by that process to be traced as well. If **-Pp** is specified, all processes and threads subsequently created by that process are traced. If **-t all** was specified to trace all threads, **-P** is ignored. The **-P** flag is only valid for trace channel `0`; the **-P** and **-g** flags are incompatible.

-r *reglist* Optional, and only valid for a **trace** run on a 64-bit kernel. *reglist* items are separated by commas, or enclosed in quotation marks and separated by blanks. Up to 8 registers may be specified. Valid *reglist* values are:

PURR - The PURR
Register for this cpu

MCR0, MCR1, MCRA - the MCR
Registers, 0, 1, and A

PMC1, PMC2, ... PMC8 - PMC
Registers 1 through 8.

Note: Not all registers are valid for all processors.

-s Stops tracing when the trace log fills. The **trace** daemon normally wraps the trace log when it fills up and continues to collect trace data. During asynchronous operation, this flag causes the **trace** daemon to stop trace data collection. (During interactive operation, the **quit** subcommand must be used to stop trace.)

-t *thread-id*[,*thread-id*] Traces only the listed threads. A thread ID is a decimal number. Multiple thread IDs can be separated by commas or enclosed in quotes and separated by spaces.

Also, the thread list can be `all` or `*`, indicating that all threads are to be traced. This is useful for tracing all thread-related events without tracing interrupt-related events. However, if **-t all** and **-l** are both specified, this is the same as specifying neither one; all events are traced. Another way to say this is that **trace** and **trace -lt all** are identical.

The **-t** flag is only valid for trace channel `0`, the **-t** and **-g** flags are incompatible.

-T *Size*

Overrides the default trace buffer size of 128 KB with the value stated. You must be root to request more than 1 MB of buffer space. The maximum possible size is 268435184 bytes, unless the **-f** flag is used, in which case it is 536870368 bytes. The smallest possible size is 8192 bytes, unless the **-f** flag is used, in which case it is 16392 bytes. Sizes between 8192 and 16392 will be accepted when using the **-f** flag; however, the actual size used will be 16392 bytes.

> **Note:** In the circular and the alternate modes, the trace buffer size must be one-half or less the size of the trace log file. In the single mode, the trace log file must be at least the size of the buffer. See the **-L** flag for information on controlling the trace log file size. Also note that trace buffers use pinned memory, which means they are not pageable. Therefore, the larger the trace buffers, the less physical memory is available to applications.
>
> Unless the **-b** or **-B** flags are specified, the system attempts to allocate the buffer space from the kernel heap. If this request can not be satisfied, the system then attempts to allocate the buffers as separate segments.

The **-f** flag actually uses two buffers, which behave as a single buffer (except that a buffer wraparound trace hook will be recorded when the first buffer is filled).

-x *program-specification*

Traces the specified program. The *program-specification* specifies a program and parameters as they would be when running the program from the shell, except that the program specification must be in quotes if more than just the program's name is given. The trace is stopped automatically when the program exits, and returns the program's return code. By default, any processes and threads created by the program are also traced; as if **-Pp** was specified. To change this behavior, use **-Pn** to specify no trace propagation, or **-Pt** to propagate trace only to threads created by the program's original process.
Note: The **-x** flag implies asynchronous tracing, as if the **-a** flag had also been specified.

-X *program-specification*

The **-X** flag works like the -x flag, except that the trace is not automatically stopped when the program exits. This is useful for tracing programs which fork processes, and then terminate, and you want these new processes traced as well.

Subcommands

When run interactively, trace recognizes the following subcommands:

trcon	Starts the collection of trace data.
trcoff	Stops the collection of trace data.
q or **quit**	Stops the collection of trace data and exits **trace**.
! *Command*	Runs the shell command specified by the *Command* parameter.
?	Displays the summary of **trace** subcommands.

Signals

The **INTERRUPT** signal acts as a toggle to start and stop the collection of trace data. Interruptions are set to **SIG_IGN** for the traced process.

Examples

1. To use trace interactively, enter `trace`, (the > prompt is displayed), then specify the subcommands you want. For example, to trace system events during the run of the *anycmd* command, enter:

   ```
   trace
   > !anycmd
   > q
   ```

2. To avoid delays when the command finishes, start trace asynchronously (**-a**), using only one command line. Enter:

   ```
   trace -a; anycmd; trcstop
   ```

3. To trace the system itself for a period of 10 seconds, enter:

   ```
   trace -a; sleep 10; trcstop
   ```

4. To output trace data to a specific trace log file (instead of the **/var/adm/ras/trcfile** default trace log file), :

   ```
   trace -a -o /tmp/my_trace_log; anycmd; trcstop
   ```

5. To capture the execution of a **cp** command, excluding specific events from the collection process:

   ```
   trace -a -k "20e,20f" -x "cp /bin/track /tmp/junk"
   ```

   In the example above, the **-k** option suppresses the collection of events from the **lockl** and **unlockl** functions (20e and 20f events).

   Also notice that the **-x** flag was used, so only hooks associated with the **cp** command process will be traced, and no interrupt activity will be traced.

6. To trace hook 234 and the hooks that will allow you to see the process names, use:

   ```
   trace -a -j 234 -J tidhk
   ```

   This traces the hooks in the event-group ″tidhk″ plus hook 234.

7. To have trace use one set of buffers per CPU, specify:

   ```
   trace -aC all
   ```

   The files produced are **/var/adm/ras/trcfile**, **/var/adm/ras/trcfile-0**, **/var/adm/ras/trcfile-1**, etc. up to **/var/adm/ras/trcfile-**(*n*-1), where *n* is the number of CPUs in the system.

   Note: `trace -aCall -o mylog` produces the files mylog, mylog-0, mylog-1, ...

8. To trace a program that starts a daemon process, and to continue tracing the daemon after the original program has finished, use

   ```
   trace -X "mydaemon"
   ```

   The trace must be stopped with **trcstop**.

9. To trace *mydaemon*, which is currently running, use:

   ```
   trace -A mydaemon-process-id -Pp
   ```

   Where *mydaemon-process-id* is the process for *mydaemon* as returned by the **ps** command. The **-Pp** flag tells trace to also trace any processes and threads created by *mydaemon* while the trace is running.

10. To capture the PURR, and PMC1 and PMC2, type

    ```
    trace -ar "PURR PMC1 PMC2"
    ```

Files

/usr/include/sys/trcmacros.h	Defines **trchook** and **utrchook** macros.
/var/adm/ras/trcfile	Contains the default trace log file.

Related Information

The **ctctrl** command, **trcevgrp** command, the **trcnm** command, the **trcrpt** command, the **trcstop** command.

The **trchook** subroutine, **trcgen** subroutine, **trcstart** subroutine, **trcon** subroutine, **trcoff** subroutine, **trcstop** subroutine.

The **trcgenk** kernel service.

Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*

Performance Analysis with the Trace Facility in *Performance management*.

Debug and Performance Tracing in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

---

traceroute Command

Purpose

Prints the route that IP packets take to a network host.

Syntax

traceroute [**-m** *Max_ttl*] [**-n**] [**-p** *Port*] [**-q** *Nqueries*] [**-r**] [**-s** *SRC_Addr*] [**-t** *TypeOfService*] [**-v**] [**-w** *WaitTime*] *Host* [*PacketSize*]

Description

> **Attention:** The **traceroute** command is intended for use in network testing, measurement, and management. It should be used primarily for manual fault isolation. Because of the load it imposes on the network, the **traceroute** command should not be used during normal operations or from automated scripts.

The **traceroute** command attempts to trace the route an IP packet follows to an Internet host by launching UDP probe packets with a small maximum time-to-live (*Max_ttl* variable), then listening for an ICMP **TIME_EXCEEDED** response from gateways along the way. Probes are started with a *Max_ttl* value of one hop, which is increased one hop at a time until an ICMP **PORT_UNREACHABLE** message is returned. The ICMP **PORT_UNREACHABLE** message indicates either that the host has been located or the command has reached the maximum number of hops allowed for the trace.

The **traceroute** command sends three probes at each *Max_ttl* setting to record the following:

- *Max_ttl* value
- Address of the gateway
- Round-trip time of each successful probe

The number of probes sent can be increased by using the **-q** flag. If the probe answers come from different gateways, the command prints the address of each responding system. If there is no response from a probe within a 3-second time-out interval, an * (asterisk) is printed for that probe.

The **traceroute** command prints an ! (exclamation mark) after the round-trip time if the *Max_ttl* value is one hop or less. A maximum time-to-live value of one hop or less generally indicates an incompatibility in the way ICMP replies are handled by different network software. The incompatibility can usually be resolved by doubling the last *Max_ttl* value used and trying again.

Other possible annotations after the round-trip notation are:

!H Host unreachable
!N Network unreachable
!P Protocol unreachable
!S Source route failed
!F Fragmentation needed

If the majority of probes result in an error, the **traceroute** command exits.

The only mandatory parameter for the **traceroute** command is the destination host name or IP number. The **traceroute** command will determine the length of the probe packet based on the Maximum Transmission Unit (MTU) of the outgoing interface. The UDP probe packets are set to an unlikely value so as to prevent processing by the destination host.

Flags

-m *Max_ttl* Sets the maximum time-to-live (maximum number of hops) used in outgoing probe packets. The default is 30 hops (the same default used for TCP connections).

-n Prints hop addresses numerically rather than symbolically and numerically. This flag saves a name-server address-to-name lookup for each gateway found on the path.

-p *Port* Sets the base UDP port number used in probes. The default is 33434. The **traceroute** command depends on an open UDP port range of *base* to *base* + *nhops* - 1 at the destination host. If a UDP port is not available, this option can be used to pick an unused port range.

-q *Nqueries* Specifies the number of probes the **traceroute** command sends at each *Max_ttl* setting. The default is three probes.

-r Bypasses the normal routing tables and sends the probe packet directly to a host on an attached network. If the specified host is not on a directly attached network, an error is returned. This option can be used to issue a **ping** command to a local host through an interface that is not registered in the **routed** daemon's routing table.

-s *SRC_Addr* Uses the next IP address in numerical form as the source address in outgoing probe packets. On hosts with more than one IP address, the **-s** flag can be used to force the source address to be something other than the IP address of the interface on which the probe packet is sent. If the next IP address is not one of the machine's interface addresses, an error is returned and nothing is sent.

-t *TypeOfService* Sets the *TypeOfService* variable in the probe packets to a decimal integer in the range of 0 to 255. The default is 0. This flag can be used to investigate whether different service types result in different paths. For more information, see TCP/IP Protocols in *Networks and communication management*. Useful values are **-t 16** (low delay) and **-t 8** (high throughput).

-v Receives packets other than **TIME_EXCEEDED** and **PORT_UNREACHABLE** (verbose output).

-w *WaitTime* Sets the time (in seconds) to wait for a response to a probe. The default is 3 seconds.

Parameters

Host Specifies the destination host, either by host name or IP number. This parameter is required.

PacketSize Specifies the probe datagram length. The default packet size is determined by the **traceroute** command based on the MTU of the outgoing interface.

Examples

1. A sample use and output is:

```
[yak 71]% traceroute nis.nsf.net.
traceroute to nis.nsf.net (35.1.1.48), 30 hops max, 56 byte packet
 1 helios.ee.lbl.gov (128.3.112.1) 19 ms 19 ms 0 ms
 2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
 3 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 39 ms 19 ms
 4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 39 ms
 5 ccn-nerif22.Berkeley.EDU (128.32.168.22) 39 ms 39 ms 39 ms
 6 128.32.197.4 (128.32.197.4) 40 ms 59 ms 59 ms
 7 131.119.2.5 (131.119.2.5) 59 ms 59 ms 59 ms
 8 129.140.70.13 (129.140.70.13) 99 ms 99 ms 80 ms
 9 129.140.71.6 (129.140.71.6) 139 ms 239 ms 319 ms
10 129.140.81.7 (129.140.81.7) 220 ms 199 ms 199 ms
11 nic.merit.edu (35.1.1.48) 239 ms 239 ms 239 ms
```

Lines 2 and 3 are the same due to a bug in the kernel on the second hop system (`lbl-csam.arpa`) that forwards packets with a zero time-to-live. Host names are not printed in lines 6 through 10 because the National Science Foundation Network (NSFNet, `129.140`) does not provide address-to-name translations for its nodes.

2. Another output example might be:

```
[yak 72]% traceroute rip.Berkeley.EDU (128.32.131.22)
traceroute to rip.Berkeley.EDU (128.32.131.22), 30 hops max
 1 helios.ee.lbl.gov (128.3.112.1) 0 ms 0 ms 0 ms
 2 lilac-dmc.Berkeley.EDU (128.32.216.1) 39 ms 19 ms 39 ms
 3 lilac-dmc.Berkeley.EDU (128.32.216.1) 19 ms 39 ms 19 ms
 4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23) 39 ms 40 ms 19 ms
 5 ccn-nerif35.Berkeley.EDU (128.32.168.35) 39 ms 39 ms 39 ms
 6 csgw/Berkeley.EDU (128.32.133.254) 39 ms 59 ms 39 ms
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 rip.Berkeley.EDU (128.32.131.22) 59 ms! 39 ms! 39 ms!
```

In this example, exactly half of the 12 gateway hops (13 is the final destination) are ″missing.″ However, these hops were actually not gateways. The destination host, a Sun-3 workstation running Sun OS3.5, used the ttl from the arriving datagram as the ttl in its ICMP reply; thus, the reply timed out on the return path. Because ICMPs are not sent for ICMPs, no notice was received. The ! (exclamation mark) after each round-trip time indicates some type of software incompatibility problem. (The cause was diagnosed after the **traceroute** command issued a probe of twice the path length. The destination host was really only seven hops away.)

Related Information

The **netstat** command, **nslookup** command, **ping** command.

TCP/IP name resolution in *Networks and communication management*.

---

tracesoff Command

Purpose

Turns off tracing of a subsystem, a group of subsystems, or a subserver.

Syntax

Subsystem

tracesoff [**-h** *Host*] { **-g** *Group* | **-p** *SubsystemPID* | **-s** *Subsystem*}

Subserver

tracesoff [**-h** *Host*] **-t** *Type* [**-p** *SubsystemPID*] { **-o** *Object* | **-P** *SubserverPID*}

Description

The **tracesoff** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem to turn tracing off. Tracing is unsuccessful if the communication method for the subsystems is signals.

Note: Tracing is subsystem dependent.

Flags

-g *Group*	Specifies a group of subsystems to turn tracing off. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this trace action is requested. The local user must be running as root. The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-o *Object*	Specifies that a subserver *Object* name is to be passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to turn tracing off, or a particular instance of the subsystem to which the trace off subserver request is to be passed.
-P *SubserverPID*	Specifies that a *SubserverPID* is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to turn tracing off. The *Subsystem* name can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies a subsystem subserver to turn tracing off. The command is unsuccessful if the *Type* is not contained in the subserver object class.

Examples

To turn off the tracing of a subsystem, type:

```
tracesoff  -s tcpip
```

This turns the tracing off for the `tcpip` subsystem.

Files

/usr/bin/tracesoff	Contains the **tracesoff** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

Related Information

The **traceson** command.

The System resource controller in *Operating system and device management* gives an explanation of subsystems, subserver, and the System Resource Controller.

traceson Command

Purpose

Turns on tracing of a subsystem, a group of subsystems, or a subserver.

Syntax

Subsystem

traceson [**-h** *Host*] [**-l**] { **-g** *Group* | **-p** *SubsystemPID*| **-s** *Subsystem*}

Subserver

traceson [**-h** *Host*] [**-l**] **-t** *Type* [**-o** *Object*] [**-p** *SubsystemPID*] [**-P** *SubserverPID*]

Description

The **traceson** command sends the System Resource Controller a subsystem request packet that is forwarded to the subsystem to turn tracing on. Tracing is unsuccessful if the communication method for the subsystems is signals.

Note: Tracing is subsystem dependent.

Tracing may occur in either short or long form. When the **-l** flag is absent, the trace request is assumed to be a short trace.

Flags

-g *Group*	Specifies a group of subsystems to turn tracing on. The command is unsuccessful if the *Group* name is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this trace action is requested. The local user must be running as ″root″. The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-l	Specifies that a long trace is requested.
-o *Object*	Specifies that a subserver object is to be passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the subsystem to turn tracing on, or a particular instance of the subsystem to which the trace subserver request is to be passed.
-P *SubserverPID*	Specifies that a subserver PID is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies the subsystem to turn tracing on. The *Subsystem* name can be either the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* name is not contained in the subsystem object class.
-t *Type*	Specifies a subserver to turn tracing on. The command is unsuccessful if the *Type* is not contained in the subserver object class.

Examples

To turn on tracing of the `tcpip` subsystem on a foreign host, enter:

```
traceson  -h odin  -s tcpip
```

This turns on the tracing for the `tcpip` subsystem on the `odin` foreign host.

Files

/usr/bin/traceson	Contains the **traceson** command.
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the **AF_UNIX** socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

Related Information

The **tracesoff** command.

The System resource controller in *Operating system and device management* gives an explanation of subsystems, subservers, and the System Resource Controller.

---

trbsd Command

Purpose

Translates characters (BSD version).

Syntax

trbsd [**-c**] [**-d**] [**-s**] [**-A**] [*String1* [*String2*]]

Description

The **trbsd** command deletes or substitutes characters from standard input and then writes the result to standard output. The **trbsd** command is the BSD version of the **tr** command. The **trbsd** command performs three kinds of operations, depending on the character strings specified by the parameters and flags specified. The default value for either the *String1* or *String2* parameter is a null string.

Transforming Characters

If both the *String1* and *String2* parameters are specified and the **-d** flag is not specified, the **trbsd** command replaces each character from standard input that is specified by the *String1* parameter with the character in the same position in the *String2* parameter.

If the *String1* parameter specifies a character more than once, the character is translated into the character in the *String2* parameter that corresponds to the last occurrence of the character in the *String1* parameter.

Deleting Characters Using the -d Flag

If the **-d** flag is specified, the **trbsd** command deletes each character from standard input that is specified by the *String1* parameter.

Removing Sequences of Characters Using the -s Flag

If the **-s** flag is specified, the **trbsd** command deletes from standard input all but the first character in a sequence of two or more repetitions of any character specified by the *String2* parameter.

Both the *String1* and *String2* parameters must be specified when both the **-d** and **-s** flags are specified.

> **Note:** The **trbsd** command deletes all null characters from standard input before it begins processing.

Special Sequences for Expressing Strings

The strings contained in *String1* and *String2* parameters can be expressed using the following conventions:

C1-*C2*	Specifies the string of characters that collate between the character specified by the *C1* string and the character specified by the *C2* string, inclusive. The character specified by the *C1* string must collate before the character specified by the *C2* string.
*****Octal*	Specifies the character whose encoding is represented by the specified octal value. The octal value can be a one-, two-, or three-digit octal integer. Multibyte characters can be expressed by writing backslash-octal sequences for each byte.
\-	The \- (backslash, minus sign) specifies the minus sign character itself, without any special meaning as an escape character.

If the strings specified by the *String1* and *String2* parameters are not the same length, the **trbsd** command pads the shorter string to equal the length of the longer string. Padding is accomplished by duplicating the last character in the shorter string as many times as necessary.

Flags

-A	Performs all operations on a byte-by-byte basis using the ASCII collation order for ranges and character classes, instead of the collation order of the current locale.
-c	Specifies that the value of the *String1* parameter be replaced by the complement of that string. The complement is all of the characters in the character set of the current locale, except for the characters specified by the *String1* parameter. If the **-A** and **-c** flags are specified together, characters are complemented with respect to the set of all 8-bit character codes.
-d	Deletes each character from standard input that is contained in the *String1* parameter.
-s	Deletes from standard input all but the first character in a sequence of two or more repetitions of any character contained in the *String2* parameter.

Examples

1. To translate braces into parentheses, enter:
   ```
   trbsd '{}' '()' < textfile > newfile
   ```
   This translates each { (left brace) to ((left parenthesis) and each } (right brace) to) (right parenthesis). All other characters remain unchanged.
2. To interchange plus signs with minus signs, and slash characters with asterisks, enter:
   ```
   trbsd '+\-/*' '\-+*/' < textfile > newfile
   ```
   The minus sign must be entered with a backslash escape character.
3. To translate lowercase characters to uppercase, enter:
   ```
   trbsd 'a-z' 'A-Z' < textfile > newfile
   ```
4. To create a list of words in a file, enter:
   ```
   trbsd -cs 'a-zA-Z' '\012' < textfile > newfile
   ```
   This translates each sequence of characters other than lowercase letters and uppercase letters into a single newline character. The octal value 012 is the code for the newline character.
5. To replace every sequence of one or more newlines with a single newline, enter:
   ```
   trbsd -s '\012'  < textfile > newfile
   ```

Files

/usr/bin/trbsd	Contains the **trbsd** command.

/usr/ucb/tr Contains a symbolic link to the **trbsd** command.

Related Information

The **ed** command, **tr** command.

National Language Support Overview in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

---

trcctl Command

Purpose

Changes and displays system trace parameters.

Syntax

trcctl [**-d** *Directory* **-l** **-L** *LogfileSize* **-M** *LMT_log_dir* **-N** *NonrootUserBufferMax* **-o** *Logfile* **-r** **-T** *BufferSize*]

Description

The **trcctl** command will display or change the system trace default parameters. If the **-l** option (or no parameter) is specified, **trcctl** will show the values as follows:

```
Default Buffer Size: 131072
Default Log File Size: 1310720
Default Log File: /var/adm/ras/trcfile
Non-root User Buffer Size Maximum: 1048576
Default Components Directory File: /var/adm/ras/trc_ct
Default LMT Log Dir: /var/adm/ras/mtrcdir
```

Note that the default buffer and log file sizes initially depend upon the kernel. However, once they are set using this command, the effected value is the same for both kernels. The other parameters allow these default values to be changed. To change a default value, the user must be a member of the system group. Many of the flags used with **trcctl** correspond to those used by the **trace** daemon.

Flags

-d *Directory*	Specifies the default Component Trace log directory path. The default value is **/var/adm/ras/trc_ct**.
-l	Lists the current values.
-L *Value*	Specifies the default log file size. The original default value is 1310720 bytes for the 32-bit kernel, and 2621440 bytes for the 64-bit kernel. If specified with **-L**, the default will apply to both kernels.
-M *LMT_log_dir*	Specifies the default Lightweight Memory Trace log directory path. The default value is **/var/adm/ras/mtrcdir**.
-N *Value*	Specifies the maximum buffer size a non-root user may specify. The default is 1 MB, 1048576 bytes.
-o *Path*	Specifies the default log file path. The default value is **/var/adm/ras/trcfile**.
-r	Restore original default values.
-T *Value*	Specifies the default trace buffer size. The original default values are 128 KB and 256 KB for a 32- or 64-bit kernel. If specified with **-T**, the default will apply to both kernels.

Parameters

If you use 'k', 'm', or '#k', '#m' as parameters for the **-N**, **-L**, and **-T** options, **trcctl** will translate these into their respective byte totals.

k = 1024

m = 1048576

Using only 'k' or 'm', **trcctl** assumes you mean 1 kilobyte or 1 megabyte respectively. This way a root user can execute :

```
trcctl -L 10m -N m -T 256k
```

Security

The user must be a member of the system group.

Related Information

The **trace** daemon.

The **ctctrl**, **trcrpt**, **trcevgrp**, **traceon**, and **traceoff** commands.

---

trcdead Command

Purpose

Extracts trace buffers from a system dump image.

Syntax

trcdead [**-1 -2 -3 ... -7**] [**-c**] [**-M**] [**-o***Name*] *DumpImage*

Description

If the system halts while trace facilities are active, the contents of the internal trace buffers are captured in the system dump. The **trcdead** command can then be used to extract the eight active system trace channels, all Component Trace buffers, and the Lightweight Memory Trace buffers from the system dump. System trace channel 0 is extracted when no flags are provided. A system trace channel other than channel 0 is identified through a *-channelnum* flag. Component Trace buffers are identified through the **-c** flag. Lightweight Memory Trace buffers are identified through the **-M** flag. Only one type of trace buffer, or one specific system trace channel, can be extracted at one time.

The **-o** flag can be used to indicate that the extracted buffers should be written to a nondefault trace log file or directory. System trace channels are extracted to a trace log file. Component Trace buffers and Lightweight Memory Trace buffers are extracted to a directory. If the **-o** flag is not chosen, the **trcdead** command writes to the default trace log file or directory. The default log file name and directory names can be viewed and modified using the **trcctl** command.

Prior to AIX 5.3, if multicpu trace buffering was requested for multiple CPUs using **trace -C**, the associated buffers were not in the dump.

Use the **trcrpt** command to format a report from the trace log file or files.

Flags

–1, ..., -7	Retrieves the trace buffer entries for channel 1, 2, 3, 4, 5, 6, and 7. The default is channel 0.
-c	Extracts all buffers of all active Component Trace components.
-M	Extracts the Lightweight Memory Trace buffers.
-o*Name*	Specifies the file or directory (**-c**, **-M**) to which data is written.

Parameter

DumpImage	Specifies the dump image to operate on.

Examples

Note: To determine which example is more appropriate for your system, use the **sysdumpdev** command to display the current dump device assignments.

1. To extract the system trace buffer to the file named `trace_extract` from a dump located at **/var/adm/ras/dumpfile**, enter:

   ```
   trcdead -o trace_extract /var/adm/ras/dumpfile
   ```

2. To extract the system trace buffer from a dump image written to a device, enter:

   ```
   trcdead /dev/hd7
   ```

3. To extract Lightweight Memory Trace information from dump image `vmcore.0` and put it into the **/tmp** directory, enter:

   ```
   trcdead -o /tmp -M vmcore.0
   ```

Files

/**usr**/**bin**/**trcdead**	Contains the **trcdead** command.
/var/adm/ras/dumpfile	Contains the default system dump file.
/var/adm/ras/trcfile	Contains the default system trace log.
/var/adm/ras/trc_ct	Contains the default Component Trace logs.
/var/adm/ras/mtrcdir	Contains the default Lightweight Memory Trace logs.

Related Information

The **errdead** command, **sysdumpdev** command, **trcnm** command, **trcrpt** command.

The **trace** daemon.

The Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

trcevgrp Command

Purpose

Manipulates trace event groups.

Syntax

List event groups

trcevgrp -l [*event-group* [...]]

Remove event groups

trcevgrp -r [*event-group* [...]]

Add an event group

trcevgrp -a -d *"group-description"* **-h** *"hook-list"* *event-group*

Update an event group

trcevgrp -u [**-d** *"group-description"*] [**-h** *"hook-list"*] *event-group*]

Description

The **trcevgrp** command is used to maintain the trace event groups. You must be in the system group to add, delete, or change trace event groups. You *cannot* modify or delete event groups whose type is `reserved`.

Flags

-a [**-d** *group-description* | **-h** *hook-list*]	Creates a new event group. Only one event group name can be specified. Both **-d** *description* and **-h** *hook-list* must be specified when using the **-a** flag. If either **-d** or **-h** is not specified, an error is produced.
-d *group-description*	Designates the hook description. A description is required for all new groups.
-h *hook-list*	The hook list consists of trace hook IDs. The **-h** flag is required when using the **-a** flag. When updating an event group (**-u** flag), the hook-list, if specified, must contain all hook IDs for the group. List parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.
-l *event-group*	The specified group(s) are listed to standard output. If no event group is specified, all are listed. The format of the listing is as follows: *group name - group-description (type) "*hook list*"* For example: • `-l tidhk - Hooks needed to display thread name (reserved) "106,10C,134,139,465"` • `-l gka - GENERAL KERNEL ACTIVITY (files,execs,dispatches) (reserved) "106,10C,134,139,465,107,135,15b,12e,116,117,200,20E,20F"`
-r *event-group*	Removes the specified event-groups.
-u [**-d** *"group-description"* | **-h** *"hook-list"*] *event-group*	Used to update the information for an event-group. Either **-d** *description* or **-h** *hook-list* must be specified.

Examples

1. To get a listing of all event groups, type:

   ```
   trcevgrp -l
   ```

2. To add a new group, type:

   ```
   trcevgrp -a -d my group description -h "500,501,502" mygrp
   ```

   This will add the group called `mygrp`, give it the description `my group description`, and will have hooks of `500`, `501`, and `502`.

3. To add another hook to `mygrp`, type:

   ```
   trcevgrp -u -d my group description -h "500,501,502,503" mygrp
   ```

   Note: It is necessary to specify all hook IDs.

   Files the event groups are currently kept in the SWserveAt ODM database.

Related Information

The **trcdead** command, **trcnm** command, **trcrpt** command, **trcstop** command, and **trcupdate** command.

The **trace** daemon.

The **trcfmt** file format.

Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

trcnm Command

Purpose

Generates a kernel name list.

Syntax

trcnm [**-a** [*FileName*]] | [*FileName*] | **-K***Symbol1* ...

Description

The **trcnm** command generates a kernel name list used by the **trcrpt** command. A kernel name list is composed of a symbol table and a loader symbol table of an object file. The **trcrpt** command uses the kernel name list file to interpret addresses when formatting a report from a trace log file. For more information, see the **trcrpt -n** command.

If the *FileName* parameter is not specified, the default *FileName* is /unix.

Flags

-a	Writes all loader symbols to standard output. The default is to write loader symbols only for system calls.
-K*Symbol...*	Obtains the value of all command line symbols through the **knlist** command system call.

Examples

1. To obtain the value of the symbols in /unix, enter:

   ```
   trcnm -K environ errno
   ```

   This command sequence displays the following:

   ```
   environ 2FF7FFF8
   errno 2FF7FFFC
   ```

2. To print a symbol table for system calls, enter:

   ```
   trcnm
   ```

   A list similar to the following is generated:

   ```
   pin_obj_start    00000000
   header_offset    00000008
   ram_disk_start   0000000C
   ram_disk_end     00000010
   dbg_avail        00000014
   base_conf_start  00000018
   base_conf_end    0000001C
   base_conf_disk   00000020
   pin_com_start    00000024
   start            00000028
   ipl_cb           00000028
   ...
   ```

Files

/var/adm/ras/trcfile	Contains the default log file.
/tlo-tvl2/trcnam	Contains the **trcnm** command.

/etc/trcfmt Contains the trace format file.

Related Information

The **trcdead** command, **trcrpt** command, **trcstop** command, **trcupdate** command.

The **trace** daemon.

The **trcfmt** file format.

Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

trcrpt Command

Purpose

Formats a report from the trace log.

Syntax

trcrpt [**-c**] [**-C** [*CPUList* | **all**]] [**-d** *List*] [**-D** *Event-group-list*] [**-e** *Date*] [**-G**] [**-h**] [**-j**] [**-k** *List*] [**-K** *Group-list*] [**-m**] [**-n** *Name*] [**-o** *File*] [**-p** *List*] [**-r**] [**-s** *Date*] [**-t** *File*] [**-T** *List*] [**-v**] [**-O** *Options*] [**-x**] [**-M common** | **rare** | **all**[:*LMT_dir*]] [**-l** *ComponentList* | **all**[:*CT_dir*]] [*FileOrDirectory*]

Description

The **trcrpt** command reads the trace log specified by the **-M**, **-l** and *File* or *Directory* parameters, formats the trace entries, and writes a report to standard output. The default file from which the system generates a trace report is the **/var/adm/ras/trcfile** file, but you can specify an alternate log file using the **-M**, **-l** and *File* or *Directory* parameters. You can specify one or more files or directories. If you specify a file, it must be a valid trace log file, which is any file that is produced by a trace-related command. If you specify a directory, it must contain a component trace master file. If you specify the **-m** flag, all specified traces will be merged in chronological order.

You can use the System Management Interface Tool (SMIT) to run the **trcrpt** command by entering the SMIT fast path:

```
smit trcrpt
```

Flags

-c Checks the template file for syntax errors.

-C [*CPUList* | **all**] Generates a report containing events that occur on the CPUs specified. The CPUs can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. To report on all CPUs, specify `trace -C all`. The **-C** flag is not necessary unless you want to see only a subset of the CPUs traced, or have the CPU number show up in the report. If **-C** is not specified, and the trace is a multi-CPU trace, **trcrpt** generates the trace report for all CPUs, but the CPU number is not shown for each hook unless you specify `-O cpuid=on`.

-d *List* Limits report to hook IDs specified with the *List* variable. The *List* parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.

-D *Event-group-list* Limits the report to hook IDs in the *Event groups list*, plus any hook IDs specified with the **-d** flag. List parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. *Event groups* are described in Debug and Performance Tracing.

-e *Date*
Ends the report time with entries on, or before, the specified date. The *Date* variable has the form *mmddhhmmssyy* (month, day, hour, minute, second, and year). Date and time are recorded in the trace data only when trace data collection is started and stopped. If you stop and restart trace data collection multiple times during a trace session, date and time are recorded each time you start or stop a trace data collection. Use this flag in combination with the **-s** flag to limit the trace to data collected during a certain time interval.
Note: The **-e** and **-s** flags are only valid for trace log files collected without the **trace -C** flag.

-G
List all event groups. The list of groups, the hook ids in each group, and each group's description is listed to standard output.

-h
Omits the header information from the trace report and writes only formatted trace entries to standard output.

-j
Displays the list of hook IDs. The **trcrpt -j** command can be used with the **trace -j** command that includes IDs of trace events or the **trace -k** command that excludes IDs of trace events.

-k *List*
Excludes from the report hook IDs specified with the *List* variable. The *List* parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.

-K *Event-group-list*
Excludes from the report hook IDs in the *event-groups* list, plus any hook IDs specified with the **-k** flag. List parameter items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks. Event groups are described in Debug and Performance Tracing.

-l *ComponentList*
Generates a report for a multi-component trace with **ctctrl -D** or **trcdead**. The components can be separated by commas, or enclosed in double quotation marks and separated by commas or blanks. The **-l** flag is not necessary unless you want to see only a subset of the components traced. If **-l** is not specified, the command assumes the trace is a multi-component trace if a directory is given as input on the command line. Multi-component trace log files not in the default directory must either have their directory specified on the command line or with the *CT_dir* parameter in conjunction with the **-l** flag. The **-l all** option can be used to select all available components. Multiple **-l** flags can be used to specify components in different directories.

-m
Merges all specified trace files based on time stamps. Files merged from another partition, system or from two or more separate boots of the same system will produce unpredictable results. Without the **-m** flag, reports for each log file are appended to the specified output file.

-M common | **rare** | **all**[:*LMT_dir*]
Generates a report from the LMT log files obtained via the **mtrcsave** or **trcdead** command.

Use the **common** keyword if you only want events from the common LMT buffers to be reported; use the **rare** keyword if you only want events from the rare LMT buffers to be reported; use the **all** keyword if you want common and rare events to be reported.

This flag searches only the default LMT log directory unless the *LMT_dir* parameter is specified. With this parameter, the **trcrpt** command will search for the LMT files in the specified directory rather than the default LMT log directory. To merge common and rare buffers you must use the **all** keyword and the **-m** flag. The **-M** flag can only appear once.

-n *Name*
Specifies the kernel name list file to be used to interpret address for output. Usually, this flag is used when moving a trace log file to another system.

-o *File*
Writes the report to a file instead of to standard output.

-O *Options*

Specifies options that change the content and presentation of the **trcrpt** command. Arguments to the options must be separated by commas or enclosed in double quotation marks and separated by commas or spaces. Valid options are:

2line=[on|off]
Uses two lines per trace event in the report instead of one. The default value is **off**.

component=[on|off]
Displays the full component name in the trace report. The default value is **off**.

cpuid=[on|off]
Displays the physical processor number in the trace report. The default value is **off**.

endtime=*Seconds*
Displays trace report data for events recorded before the seconds specified. Seconds can be given in either an integral or rational representation. If this option is used with the **starttime** option, a specific range can be displayed.

exec=[on|off]
Displays exec path names in the trace report. The default value is **off**.

filename=[on|off]
Displays the file name from which an event was retrieved. The file name will be truncated from the left if it exceeds 40 characters. The default value is **off**.

hist=[on|off]
Logs the number of instances that each hook ID is encountered. This data can be used for generating histograms. The default value is **off**. This option cannot be run with any other option.

ids=[on|off]
Displays trace hook identification numbers in the first column of the trace report. The default value is **on**.

pagesize=*Number*
Controls the number of lines per page in the trace report and is an integer within the range of 0 through 500. The column headings are included on each page. No page breaks are present when the default value of 0 is set.

pid=[on|off]
Displays the process IDs in the trace report. The default value is **off**.

reportedcpus=[on | off]
Displays the number of CPUs remaining. This option is only meaningful for a multi-CPU trace, `trace -C`. For example, if you are reading a report from a system with 4 CPUs, and the reported CPU's value goes from 4 to 3, then you know that there are no more hooks to be reported for that CPU.

PURR=[on | off]

Tells **trcrpt** to show the PURR along with any timestamps. The PURR is displayed following any timestamps.

If the PURR is not valid for the processor traced, the elapsed time is shown instead of the PURR. If the PURR is valid, or the **cpuid** is unknown, but wasn't traced for a hook, the PURR field contains asterisks (*).

-0 *Options (Continued)*

removedups=[on | off]
Enables duplicate event detection. A count in the DUPS column displays the number of events that each event in the report represents. If this option is set to **off**, duplicate event detection will be disabled. The default value is **on**. This option is only valid when merging log files via the **-m** flag. Duplicate entries can only be detected when the CPU ID is known from the trace entry itself, not when it must be inferred. The CPU ID can be obtained from the entry in the following cases:

- A lightweight memory trace
- A multi-CPU system trace, where the **trace -C** command option was used
- A 64-bit system trace initiated with the **-p** option
- A 64-bit component trace.

starttime=*Seconds*
Displays trace report data for events recorded after the seconds specified. The specified seconds are from the beginning of the trace file. Seconds can be given in either an integral or rational representation. If this option is used with the **endtime** option, a specific range of seconds can be displayed.

svc=[on|off]
Displays the value of the system call in the trace report. The default value is **off**.

tid=[on|off]
Displays the thread ID in the trace report. The default value is **off**.

timestamp=[0|1|2|3]
Controls the reporting of the time stamp associated with an event in the trace report. The possible values are:

0 Time elapsed since the trace was started and delta time from the previous event. The elapsed time is in seconds and the delta time is in milliseconds. Both values are reported to the nearest nanosecond. This is the default.

1 Short elapsed time. Reports only the elapsed time (in seconds) from the start of the trace. Elapsed time is reported to the nearest microsecond.

2 Microsecond delta time. This is like 0, except the delta time is in microseconds, reported to the nearest microsecond.

3 No time stamp.

-p *List*
Reports the process IDs for each event specified by the *List* variable. The *List* variable may be a list of process IDs or a list of process names. List items that start with a numeric character are assumed to be process IDs. The list items can be separated by commas or enclosed in double quotation marks and separated by commas or blanks.

-r
Outputs unformatted (raw) trace entries and writes the contents of the trace log to standard output one entry at a time. Use the **-h** flag with the **-r** flag to exclude the heading. To get a raw report for CPUs in a multicpu trace, use both the **-r** and **-C** flags.

-s *Date* — Starts the report time with entries on, or before, the specified date. The *Date* variable has the form *mmddhhmmssyy* (month, day, hour, minute, second, and year). Date and time are recorded in the trace data only when trace data collection is started and stopped. If you stop and restart trace data collection multiple times during a trace session, date and time are recorded each time you start or stop a trace data collection. Use this flag in combination with the **-e** flag to limit the trace to data collected during a certain time interval.
Note: The **-e** and **-s** flags are only valid for trace log files collected without the **trace -C** flag.

-t *File* — Uses the file specified in the *File* variable as the template file. The default is the **/etc/trcfmt** file.

-T *List* — Limits the report to the kernel thread IDs specified by the *List* parameter. The list items are kernel thread IDs separated by commas or enclosed in double quotation marks and separated by commas or spaces. Starting the list with a kernel thread ID limits the report to all kernel thread IDs in the list. Starting the list with a **!** (exclamation point) followed by a kernel thread ID limits the report to all kernel thread IDs not in the list.

-v — Prints file names as the files are opened. Changes to verbose setting.

-x — Displays the exec path name and value of the system call.

Examples

1. To format the trace log file and print the result, enter:
   ```
   trcrpt | qprt
   ```
2. To send a trace report to the **/tmp/newfile** file, enter:
   ```
   trcrpt -o /tmp/newfile
   ```
3. To display process IDs and exec path names in the trace report, enter:
   ```
   trcrpt -O pid=on,exec=on
   ```
4. To create trace ID histogram data, enter:
   ```
   trcrpt -O hist=on
   ```
5. To produce a list of all event groups, enter:
   ```
   trcrpt -G
   ```
   The format of this report is shown under the **trcevgrp** command.
6. To generate back-to-back LMT reports from the common and rare buffers, specify:
   ```
   trcrpt -M all
   ```
7. If, in the above example, the LMT files reside at **/tmp/mydir**, and we want the LMT traces to be merged, specify:
   ```
   trcrpt -m -M all:/tmp/mydir
   ```
8. To merge the system trace with the **scdisk.hdisk0** component trace, specify:
   ```
   trcrpt -m -l scdisk.hdisk0 /var/adm/ras/trcfile
   ```
9. To merge LMT with the system trace while not eliminating duplicate events, specify:
   ```
   trcrpt -O removedups=off -m -M all /var/adm/ras/trcfile
   ```
10. To merge all component traces in **/tmp/mydir** with the LMT traces in the default LMT directory while showing the source file for each trace event, specify:
    ```
    trcrpt -O filename=on -m -M all /tmp/mydir
    ```
    Note: This is equivalent to:
    ```
    trcrpt -O filename=on -m -M all -l all:/tmp/mydir
    ```

Note: If the traces are from a 64-bit kernel, duplicate entries will be removed. However, on the 32-bit kernel, duplicate entries will not be removed since we do not know the CPU IDs of the entries in the components traces.

Files

/usr/bin/trcrpt	Contains the **trcrpt** command.
/var/adm/ras/trcfile	Contains the default log file.
/var/adm/ras/mtrcdir	Location of the default LMT dump directory
/var/adm/ras/trc_ct	Location of the default CT® dump directory.
/etc/trcfmt	Contains the trace format file.

Related Information

The **ctctrl** command, **trcctl** command, **trcdead** command, **trcevgrp** command, **trcnm** command, **trcstop** command, **trcupdate** command.

The **trace** daemon.

The **trcfmt** file format.

Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

Monitoring and tuning commands and subroutines in *Performance management*.

Debug and Performance Tracing in *AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts*.

---

trcstop Command

Purpose

Stops the trace function.

Syntax

trcstop

Description

The **trcstop** command ends a trace session.

You can use the System Management Interface Tool (SMIT) to run the **trcstop** command. To use SMIT, enter:

```
smit trcstop
```

Example

To terminate the trace background process, enter:

```
trcstop
```

File

/usr/bin/trcstop	Contains the **trcstop** command.

Related Information

The **trcrpt** command.

The **trace** daemon.

The Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

trcupdate Command

Purpose

Adds, replaces, or deletes trace report format templates.

Syntax

trcupdate [**-o**] [**-t** *File*] [**-v**] [**-x** *IDList*] [*File*]

Description

The **trcupdate** command adds, replaces, or deletes trace report format templates in the **/etc/trcfmt** or the **/etc/trcfmt.Z** file. When the **/etc/trcfmt.Z** file is used, the **trcupdate** command uncompresses the file, updates it, and recompresses it. The **trcupdate** command creates an ″undo″ file named *File***.undo.trc** in the specified directory.

The **trcupdate** command adds the extension **.trc** to the file name and reads update commands from that file. The undo file is input to the **trcupdate** command if the **-o** (override) flag is specified. When the **-o** flag is specified, the **trcupdate** command undoes the changes previously made to the file.

The first field of each template contains an operator:

+ The plus sign indicates that a template is to be added or replaced. The field that follows this operator contains the template to be replaced.

- The minus sign indicates that a template is to be deleted. The field after this operator contains the hook ID of the template to delete. Operations are performed in the order in which they appear.

The input to the **trcupdate** command must contain the following as the first line:

```
* /etc/trcfmt
```

The following is a sample trace file:

```
* /etc/trcfmt
+ 15A 1.0 new_fmt
- 1B3
- A14
```

When adding or replacing, the **trcupdate** command compares the version numbers of each input template with the version number of the template with the same hook ID. If the version number of the input template is later than the version of the existing template, the **trcupdate** command replaces the old template with the input template. If a template does not exist, then the input template is added to the file.

The **trcupdate** command will not modify the **/etc/trcfmt** file if a syntax error is detected in the update file.

Flags

-o Overrides the old template with the input template without verifying the version number of either template.

-t *File*	Specifies a file, instead of the **/etc/trcfmt** or the **/etc/trcfmt.Z** file, to be used as the template file.
-v	Prints the file names as each file is opened.
-x *IDList*	Extracts the templates specified in the *IDList* from the template file and writes them to standard output. The *IDList* parameter lists the hook IDs.

Security

Access Control: None, but you must have write authority to the template file you are changing. The default is **/etc/trcfmt**.

Examples

1. To add a template, enter:

```
trcupdate
* /etc/trcfmt
+ 15A 1.0 new_fmt
```

2. To delete a template, enter:

```
trcupdate
* /etc/trcfmt
- 15A 1.0 new_fmt
```

3. To replace a template, enter:

```
trcupdate
* /etc/trcfmt
+ 15A 1.0 new_fmt
```

Files

/usr/bin/trcupdate	Contains the **trcupdate** command.
/etc/trcfmt	Contains the trace format file.
/usr/include/sys/trcmacros.h	Defines **trchook** and **utrchook** macros.

Related Information

The **trcdead** command, **trcrpt** command.

The **trace** daemon.

The **trcfmt** file format.

The Trace Facility Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

troff Command

Purpose

Formats text for printing on typesetting devices.

Syntax

troff [**-a**] [**-i**] [**-q**] [**-z**] [**-F** *Directory*] [**-n** *Number*] [**-o** *List*] [**-r** *ANumber*] [**-s** *Number*] [**-T** *Name*] [**-mm** | **-me** | **-mptx** | **-ms** | **-man** | **-mv**] [**-M** *Media*] [*File* ... | **-**]

Description

The **troff** command reads one or more files and formats the text for printing on a phototypesetter or comparable device. A postprocessor is then required to post process the output of the **troff** command to the target device. See the accompanying example.

If no file is specified or the **-** (minus) flag is not the last parameter, standard input is read by default.

For the 3812, 3816, and Hewlett-Packard LaserJet Series II printer, the default fonts are the native fonts for the printer. Additional fonts also are available for these printers, which can be loaded through the use of the **troff .fp** directive. These fonts are stored on the host in the directory **/usr/lib/font/dev**Printer/**bitmaps**, and downloaded to the printer as necessary.

Typefaces

Three different typefaces are provided in four styles. The following chart shows the relationship between typeface, style, and the name that the **troff** command uses to access the font.

Note: The fonts in this set are based on the Computer Modern letter forms developed by Donald E Knuth. (Refer to Knuth, Donald: *Computer Modern Typefaces*. Addison-Wesley, 1986.)

Typeface	Regular	Italic	Bold	Italic
Roman	cr	cR	Cr	CR
Sans Serif	cs	cS	Cs	CS
Typewriter	ct	cT	Ct	CT
troff special	sp			

These fonts are all provided in the standard 15 troff sizes: 6, 7, 8, 9, 10, 11, 12, 14, 16, 28, 20, 22, 24, 28, and 36 points.

For example, `.fp 1 Cr` loads the Roman bold font into position 1.

Note: The **.tl** request cannot be used before the first break-producing request in the input to the **troff** command.

Flags

-a Sends a printable ASCII approximation of the results to standard output.

-F*Directory* Accesses font information from the Directory/**dev**Name directory instead of the default **/usr/lib/font/dev**Name directory (where *Name* is specified by the **-T** flag).

-i Reads standard input after there are no more files.

-M *Media* Specifies a paper size in order to determine the amount of imageable area on the paper. Valid values for the *Media* variable are:

A4 Specifies a paper size of 8.3 X 11.7 inches (210 X 297 mm).

A5 Specifies a paper size of 5.83 X 8.27 inches (148 X 210 mm).

B5 Specifies a paper size of 6.9 X 9.8 inches (176 X 250 mm).

EXEC Specifies a paper size of 7.25 X 10.5 inches (184.2 X 266.7 mm).

LEGAL
Specifies a paper size of 8.5 X 14 inches (215.9 X 355.6 mm).

LETTER
Specifies a paper size of 8.5 X 11 inches (215.9 X 279.4 mm). This is the default value.

Note: The *Media* variable is not case-sensitive.

-n*Number* Numbers the first printed page with the value specified by the *Number* variable.

-o*List* Prints only pages specified by the *List* variable, which consists of a comma-separated list of page numbers and ranges:

- A range of *Start-Stop* means print pages *Start* through *Stop.* For example: `9-15` prints pages 9 through 15.
- An initial *-Stop* means print from the beginning to page *Stop.*
- A final *Start-* means print from page*Start* to the end.
- A combination of page numbers and ranges prints the specified pages. For example: `-3,6-8,10,12-` prints from the beginning through page 3, pages 6 through 8, page 10, and page 12 to the end.
  Note: When this flag is used in a pipeline (for example, with one or more of the **pic**, **eqn,** or **tbl** commands), you might receive a `broken pipe` message if the last page in the document is not specified in the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

-q Calls the simultaneous input and output mode of the **.rd** request.

-r*ANumber* Sets the register specified by the *A* variable to the specified number. The *A* variable value must have a one-character ASCII name.

-s*Number* Generates output to make the typesetter stop every specified number of pages.

-T*Name* Prepares the output for the specified printing device. Phototypesetters or comparable printing devices use the following *Name* variables for operating system international extended characters. The default is **ibm3816**.
Note: You get a message that reads `bad point size` if your device does not support the point size that you specified. The **troff** command uses the closest valid point size to continue formatting.

canonls
Canon Lasershot LBP-B406S/D/E,A404/E,A304E.

ibm3812
3812 Pageprinter II.

ibm3816
3816 Pageprinter.

hplj Hewlett-Packard LaserJet II.

ibm5585H-T
5585-H01 Traditional Chinese Language support.

ibm5587G
5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 Kanji Printer multibyte language support.

psc PostScript printer.

X100 AIXwindows display.

Note: You also can set the **TYPESETTER** environment variable to one of the preceding values instead of using the **-T***Name* flag of the **troff** command.

-man Selects the **man** macro processing package.

-me Selects the **me** macro processing package.

-mm Selects the **mm** macro processing package.

-mptx Selects the **mptx** macro processing package.

-ms Selects the **ms** macro processing package.

-mv Selects the **mv** macro processing package.

See Macro Packages for Formatting Tools for more information on the macros.

-z Prints only messages generated by **.tm** (workstation message) requests.

- Forces input to be read from standard input.

Environment Variables

TYPESETTER Contains information about a particular printing device.

Examples

The following is an example of the **troff** command:

```
troff -Tibm3812 File | ibm3812 | qprt
```

Macro Packages for Formatting Tools

The following macro packages are part of the Formatting Tools in the Text Formatting System and are described in more detail on the next pages:

man	Enables you to create your own manual pages from online manual pages.
me	Provides macros for formatting papers.
mm	Formats documents with **nroff** and **troff** formatters.
mptx	Formats a permuted index.
ms	Provides a formatting facility for various styles of articles, theses, and books.
mv	Typesets English-language view graphs and slides by using the **troff** command.

man Macro Package for the nroff and troff Commands

The **man** macro package is provided to enable users to create their own manual pages from online manual pages that have been processed with either the **nroff** command or **troff** command. The **man** macro package is used with either the **nroff** command or the **troff** command.

Special macros, strings, and number registers exist, internal to the **man** macro package, in addition to the following lists of format macros, strings, and registers. Except for the names predefined by the **troff** command and the **d**, **m**, and **y** number registers, all such internal names are of the form *SymbolAlpha*, where *Symbol* is one of **)**, **]**, or **}**, and *Alpha*is any alphanumeric character.

The **man** macro package uses only the Roman font. If the input text of an entry contains requests for other fonts (for example, the **.I** format macro, **.RB** request, or **\fI** request) the corresponding fonts must be mounted.

Format Macros

The following macros are used to alter the characteristics of manual pages that are formatted using the **man**macro package.

Type font and size are reset to default values before each paragraph and after processing font- and size-setting macros (for example, the **.I** format macro, **.SM** format macro, and **.B** format macro).

Tab stops are neither used nor set by any of the format macros except the **.DT** format macro and the **.TH** format macro.

.B [*Text***]**

Makes text bold.

The *Text* variable represent up to six words; use “ ” (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.DT Restores default tab settings every 5 ens for the **nroff** command and every 7.2 ens for the **troff** command.

.HP [*Indent***]**
Begins a paragraph with a hanging indent as specified by the *Indent*variable.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.I [*Text***]**
Makes text italic.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.IP [*Tag***] [***Indent***]**
Same as the **.TP** *Indent* macro with the *Tag* variable; if the value of the *Tag* variable is **NULL**, begin indented paragraph. This macro is often used to get an indented paragraph without a tag.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.P Begins paragraph with normal font, point size, and indent. The **.PP** macro is a synonym for the **mm** macro package **.P** macro.

.PD [*Number***]**
Sets inter-paragraph distance the number of vertical spaces specified by the *Number* parameter. The default *Number* variable value is 0.4v for the **troff** command and 1v for the **nroff** command.

.PM [*Indicator***]**
Sets proprietary marking as follows:

Indicator	**Marking**
P	PRIVATE
N	NOTICE
No *Indicator* specified	Turns off proprietary marking.

.RE [*Number***]**
Ends relative indent (**.RS**) at indent level position specified by the *Number* variable. If the *Number* variable value is omitted, return to the most recent lower indent level.

.RI *Character1Character2...*
Concatenates the Roman *Character1* with the italic *Character2*; alternate these two fonts up to six sets of *Character1Character2*. Similar macros alternate between any two of Roman, italic, and bold: the **.IR**, **.RB**, **.BR**, **.IB**, and **.BI** macros.

.RS [*Indent***]**
Increases relative indent (initially zero). Indent all output an extra number of units from the left margin as specified by the *Indent* variable.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

.SH [*Text***]**
Places subhead text.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.SM [*Text***]**
Makes text one point smaller than default point size.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.SS [*Text***]**
Places sub-subhead text.

The *Text* variable represent up to six words; use " " (double quotation marks) to include character spaces in a word. If the variable is empty, this treatment is applied to the next input text line that contains text to be printed. For example, use the **.I** format macro to italicize an entire line, or use the **.SM** and **.B** format macros to produce an entire line of small-bold text. By default, hyphenation is turned off for the **nroff** command, but remains on for the **troff** command.

.TH [*Title***][***Section***][***Commentary***][***Name***]**
Sets the title and entry heading. This macro calls the **.DT** format macro.

Variable	Marking
Title	Title
Section	Section number
Commentary	Extra commentary
Name	New manual name.

Note: If the **.TH** format macro values contain character spaces that are not enclosed in " " (double quotation marks), irregular dots are displayed on the output.

.TP [*Indent***]**
Begins indented paragraph with hanging tag. The next input line that contains text is the tag. If the tag does not fit, it is printed on a separate line.

If the *Indent* variable is omitted, the previous *Indent* value is used. This value is set to its default (5 ens for the **nroff** command and 7.2 ens for the **troff** command) by the **.TH** format macro, **.P** format macro, and **.RS** format macro, and restored by the **.RE** format macro. The default unit for *Indent* is ens.

Strings

***R** Adds trademark, (Reg.) for the **nroff** command and the registered trademark symbol for the **troff** command.
***S** Changes to default type size.
***(Tm** Adds trademark indicator.

Registers

IN Indent left margin relative to subheads. The default is 7.2 ens for the **troff** command and 5 ens for the **nroff** command.
LL Line length including the value specified by the **IN** register.

PD Current inter-paragraph distance.

Flags

-rs1 Reduces default page size of 8.5 inches by 11 inches with a 6.5-inch by 10-inch text area to a 6-inch by 9-inch page size with a 4.75-inch by 8.375-inch text area. This flag also reduces the default type size from 10-point to 9-point and the vertical line spacing from 12-point to 10-point.

Examples

1. To process the file `your.book` and pipe the formatted output to the local line printer, `qprt`, type:
   ```
   nroff -Tlp -man your.book | qprt -dp
   ```
2. To process the files `my.book` and `dept.book`, which contain tables, and pipe the formatted output to the local line printer, `qprt`, type:
   ```
   tbl my.book dept.book | nroff -Tlp -man | col -Tlp | qprt -dp
   ```

   Note: Before the output is sent to `qprt`, it is first filtered through the **col** command to process reverse linefeeds used by the **tbl** command.
3. To process the file `group`, which contains pictures, graphs, and tables, and prepare the formatted output for processing on the IBM 3816 printer, type:
   ```
   grap group | pic | tbl | troff -Tibm3816 -man \
     | ibm3816 | qprt -dp
   ```

Notes:

1. If manual pages created with the **man** macro package are intended for an online facility, components requiring the **troff** command, such as the **grap** or **pic** command, should be avoided.
2. The **grap** command precedes the **pic** command because it is a preprocessor to the **pic** command; the reverse does not format correctly.
3. The **col** command is not required as a filter to the **tbl** command; typeset documents do not require reverse linefeeds.

me Macro Package for the nroff and troff Commands

The **me** package of the **nroff** and **troff** command macro definitions provides a formatting facility for technical papers in various formats. The **col** command may be required to postprocess **nroff** output in certain cases.

The macro requests are defined in the following section, in **me Requests**. Many **nroff**/**troff** requests can have unpredictable results in conjunction with this package. However, the following requests can be used after the first **.pp** request:

.bp	Begins new page.
.br	Breaks output line here.
.ce [*Number*]	Centers next specified number of lines. Default is 1 (one).
.ls [*Number*]	Sets line spacing. Text is single-spaced if *Number* is set to 1 (one); double-spaced if the value is set to 2.
.na	Leaves right margin unjustified.
.sp [*Number*]	Inserts the specified number of spacing lines.
.sz [*+*]*Number*	Adds the specified number to point size.
.ul [*Number*]	Underlines next specified number of lines. Default is 1 (one).

Output of the **eqn**, **neqn**, **refer**, and **tbl** commands preprocessors for equations and tables can be used as input.

me Requests

The following list contains all macros, strings, and number registers available in the **me** macros. Selected **troff** commands, registers, and functions are included.

\(space)	Defines unpaddable space (**troff** command built-in function).
\"	Comments to end of line (**troff** command built-in function).
***#**	Indicates optional delayed text tag string.
\$*Number*	Interpolates the value specified by the *Number* variable (**troff** command built-in function).
\n($0	Defines section depth (number register).
.$0	Started after section title printed (user-definable macro).
\n($1	Defines first section number (number register).
.$1	Started before printing depth 1 (one) section (user-definable macro).
\n($2	Defines second section number (number register).
.$2	Started before printing depth 2 section (user-definable macro).
\n($3	Defines third section number (number register).
.$3	Started before printing depth 3 section (user-definable macro).
\n($4	Defines fourth section number (number register).
.$4	Started before printing depth 4 section (user-definable macro).
\n($5	Defines fifth section number (number register).
.$5	Started before printing depth 5 section (user-definable macro).
\n($6	Defines sixth section number (number register).
.$6	Started before printing depth 6 section (user-definable macro).
.$C	Called at beginning of chapter (user-definable macro).
.$H	Indicates text header (user-definable macro).
\n($R	Defines relative vertical spacing in displays (number register defined by default; changing is not recommended).
\n($c	Defines current column header (number register).
.$c	Prints chapter title (macro defined by default; changing is not recommended).
\n($d	Indicates delayed text number (number register).
\n($f	Indicates footnote number (number register).
.$f	Prints footer (macro defined by default; changing is not recommended).
.$h	Prints header (macro defined by default; changing is not recommended).
\n($i	Defines paragraph base indent (number register).
\n($l	Defines column width (number register).
\n($m	Indicates number of columns in effect (number register).
***($n**	Indicates section name (string).
\n($p	Defines numbered paragraph number (number register).
.$p	Prints section heading (macro defined by default; changing is not recommended).
\n($r	Defines relative vertical spacing in text (number register defined by default; changing is not recommended).
\n($s	Defines column indent (number register).
.$s	Separates footnotes from text (macro defined by default; changing is not recommended).
\n%	Defines current page number (number register defined by default; changing is not recommended).
\&	Indicates zero-width character; useful for hiding controls (**troff** command built-in function).
\(*XX*	Interpolates special character specified by the *XX* variable (**troff** command built-in function).
.(b	Begins block (macro).
.(c	Begins centered block (macro).
.(d	Begins delayed text (macro).
.(f	Begins footnote (macro).
.(l	Begins list (macro).
.(q	Begins quote (macro).
.(x*Index*	Begins indexed item in the specified index (macro).

.(z Begins floating keep (macro).

.)b Ends block (macro).

.)c Ends centered block (macro).

.)d Ends delayed text (macro).

.)f Ends footnote (macro).

.)l Ends list (macro).

.)q Ends quote (macro).

.)x Ends index entry (macro).

.)z Ends floating keep (macro).

*****String* Interpolates the value specified by the *String* variable (**troff** command built-in function).

*****String1String2* Interpolates the value specified by the *String1String2* variable (**troff** command built-in function).

***** Indicates optional footnote tag string.

.++*mH* Macro to define paper section. The value specified by the *m* variable defines the part of the paper. The *m* variable can have the following values:

C Defines chapter.

A Defines appendix.

P Defines preliminary information, such as abstract and table of contents.

B Defines bibliography.

RC Defines chapters to be renumbered from page 1 (one) of each chapter.

RA Defines appendix to be renumbered from page 1 (one).

The *H* parameter defines the new header. If there are any spaces in it, the entire header must be quoted. If you want the header to have the chapter number in it, use the string **\\\n(ch**. For example, to number appendixes A.1, A.2, ..., type: **.**++ RA '''\\\n(ch.%'**.** Each section (such as chapters and appendixes) should be preceded by the **.+c** request.

.+c*Title* Begins chapter (or appendix, for instance, as set by the **.++** macro). The value specified by the *Title* variable is the chapter title (macro).

***,** Indicates cedilla (string).

\- Indicates minus sign (**troff** command built-in function).

***-** Indicates 3/4 em dash (string).

\0 Defines unpaddable digit-width space (**troff** command built-in function).

.1c Reverts to single-column output (macro).

.2c Begins two-column output (macro).

***:** Indicates umlaut (string).

***<** Begins subscript (string).

***>** Ends subscript (string).

.EN Ends equation. Space after equation produced by the **eqn** command or **neqn** command (macro).

.EQ*XY* Begins equation; breaks out and adds space. The value specified by the *Y* variable is the equation number. The optional *X* variable value might be any of the following:

I Indents equation (default).

L Left-adjusts equation.

C Centers equation (macro).

\L'*Distance*' Indicates vertical line-drawing function for the specified distance (**troff** command built-in function).

.PE Ends pic picture (macro).

.PF Ends pic picture with flyback (macro).

.PS Starts pic picture (macro).

.TE Ends table (macro).

.TH Ends header of table (macro).

.TS *X*	Begins table. If the value of the *X* variable is **H**, the table has a repeated heading (macro).
***[**	Begins superscript (string).
\n(.$	Defines number of options to macro (number register defined by default; changing is not recommended).
\n(.i	Indicates current indent (number register defined by default; changing is not recommended).
\n(.l	Indicates current line length (number register defined by default; changing is not recommended).
\n(.s	Indicates current point size (number register defined by default; changing is not recommended).
***(**4	Indicates acute accent (string).
***(`**	Indicates grave accent (string).
\(4	Indicates acute accent (**troff** command built-in function).
\(`	Indicates grave accent (**troff** command built-in function).
***]**	Ends superscript (string).
\^	Indicates 1/12 em narrow space (**troff** command built-in function).
***^**	Indicates caret (string).
.ac*AuthorNumber*	Sets up for ACM-style output. The *Author* variable specifies the author name or names. The *Number* variable specifies the total number of pages. Must be used before the first initialization (macro).
.ad	Sets text adjustment (macro).
.af	Assigns format to register (macro).
.am	Appends to macro (macro).
.ar	Sets page numbers in Arabic (macro).
.as	Appends to string (macro).
.b *X*	Prints in boldface the value specified by the *X* variable. If the *X* variable is omitted, boldface text follows (macro).
.ba +*Number*	Augments the base indent by the specified *Number* value. Sets the indent on regular text such as paragraphs (macro).
.bc	Begins new column (macro).
.bi *X*	Prints in bold italic the value specified by the *X* parameter, in no-fill mode only. If the *X* parameter is not used, bold italic text follows (macro).
\n(bi	Displays block indent (number register).
.bl	Requests blank lines, even at top of page (macro).
\n(bm	Sets bottom title margin (number register).
.bp	Begins page (macro).
.br	Sets break; starts new line (macro).
\n(bs	Displays block pre- or post-spacing (number register).
\n(bt	Blocks keep threshold (number register).
.bu	Begins bulleted paragraph (macro).
.bx *X*	Prints in no-fill mode only the value specified by the *X* variable in box (macro).
\c	Continues input (**troff** command built-in function).
.ce	Centers lines (macro).
\n(ch	Defines current chapter number (number register).
.de	Defines macro (macro).
\n(df	Displays font (number register).
.ds	Defines string (macro).
\n(dw	Defines current day of week (number register).
***(dw**	Defines current day of week (string).
\n(dy	Defines current day of month (number register).
\e	Indicates printable version of \ (backslash) (**troff** command built-in function).
.ef'*X*'*Y*'*Z*'	Sets even-page footer to the values specified by the *XYZ* variables (macro).
.eh'*X*'*Y*'*Z*'	Sets even-page header to the values specified by the *XYZ* variables (macro).
.el	Specifies the else part of an if/else conditional (macro).
.ep	Ends page (macro).

\n(es	Indicates equation pre- or post-space (number register).
\f*Font*	Sets inline font change to the specified *Font* variable value (**troff** command built-in function).
\f(*Fontf*	Sets inline font change to the specified *Fontf* variable value (**troff** command built-in function).
.fc	Sets field characters (macro).
\n(ff	Sets footnote font (number register).
.fi	Fills output lines (macro).
\n(fi	Indicates footnote indent, first line only (number register).
\n(fm	Sets footer margin (number register).
.fo *'X'Y'Z'*	Sets footer to the values specified by the *XYZ* variables (macro).
\n(fp	Sets footnote point size (number register).
\n(fs	Sets footnote pre-space (number register).
\n(fu	Sets footnote indent from right margin (number register).
\h'*Distance*'	Sets local horizontal motion for the specified distance (**troff** command built-in function).
.hc	Sets hyphenation character (macro).
.he *'X'Y'Z'*	Sets header to the values specified by the *XYZ* variables (macro).
.hl	Draws horizontal line (macro).
\n(hm	Sets header margin (number register).
.hx	Suppresses headers and footers on next page (macro).
.hy	Sets hyphenation mode (macro).
.i *X*	Italicizes the value specified by the *X* variable. If the *X*variable is omitted, italic text follows (macro).
.ie	Specifies the else part of an if/else conditional (macro).
.if	Designates a conditional (macro).
\n(ii	Sets indented paragraph indent (number register).
.in	Indents (transient); use the **.ba** macro if pervasive (macro).
.ip *X Y*	Starts indented paragraph, with hanging tag specified by the *X* variable. Indentation is the en value specified by the *Y* variable. Default is 5 (macro).
.ix	Indents, no break (macro).
\l'*Distance*'	Starts horizontal line-drawing function for the specified distance (**troff** command built-in function).
.lc	Sets leader repetition character (macro).
.lh	Interpolates local letterhead (macro).
.ll	Sets line length (macro).
.lo	Reads in a file of local macros of the form **.***x. Must be used before initialization (macro).
.lp	Begins left-justified paragraph (macro).
***(lq**	Designates left quotation marks (string).
.ls	Sets multi-line spacing (macro).
.m1	Sets space from top of page to header (macro).
.m2	Sets space from header to text (macro).
.m3	Sets space from text to footer (macro).
.m4	Sets space from footer to bottom of page (macro).
.mc	Inserts margin character (macro).
.mk	Marks vertical position (macro).
\n(mo	Defines month of year (number register).
***(mo**	Defines current month (string).
\n*X*	Interpolates number register specified by the *X* variable value (number register).
\n(*XX*	Interpolates number register specified by the *XX* variable (number register).
.n1	Sets number lines in margin (macro).
.n2	Sets number lines in margin (macro).
.na	Turns off text adjustment (macro).
.ne*Number*	Sets the specified number of lines of vertical space (macro).

.nf	Leaves output lines unfilled (macro).
.nh	Turns off hyphenation (macro).
.np	Begins numbered paragraph (macro).
.nr	Sets number register (macro).
.ns	Indicates no-space mode (macro).
***o**	Indicates superscript circle (such as for Norse A; string).
.of'*X*'*Y*'*Z*'	Sets odd footer to the values specified by the *XYZ* variables (macro).
.oh'*X*'*Y*'*Z*'	Sets odd header to the values specified by the *XYZ* variables (macro).
.pa	Begins page (macro).
.pd	Prints delayed text (macro).
\n(pf	Indicates paragraph font (number register).
\n(pi	Indicates paragraph indent (number register).
.pl	Sets page length (macro).
.pn	Sets next page number (macro).
.po	Sets page offset (macro).
\n(po	Simulates page offset (number register).
.pp	Begins paragraph, first line indented (macro).
\n(pp	Sets paragraph point size (number register).
\n(ps	Sets paragraph pre-space (number register).
.q	Indicates quoted (macro).
***(qa**	For all (string).
***qe**	There exists (string).
\n(qi	Sets quotation indent; also shortens line (number register).
\n(qp	Sets quotation point size (number register).
\n(qs	Sets quotation pre- or post-space (number register).
.r	Sets Roman text to follow (macro).
.rb	Sets real bold font (macro).
.re	Resets tabs to default values (macro).
.rm	Removes macro or string (macro).
.rn	Renames macro or string (macro).
.ro	Sets page numbers in Roman (macro).
***(rq**	Indicates right quotation marks (string).
.rr	Removes register (macro).
.rs	Restores register (macro).
.rt	Returns to vertical position (macro).
\s*Size*	Changes inline size to specified size (**troff** command built-in function).
.sc	Reads in a file of special characters and diacritical marks. Must be used before initialization (macro).
\n(sf	Sets section title font (number register).
.sh*LevelTitle*	Indicates section head to follow; font automatically bold. The *Level* variable specifies the level of section. The *Title* variable specifies the title of section (macro).
\n(si	Sets relative base indent-per-section depth (number register).
.sk	Leaves the next page blank. Only one page is remembered ahead (macro).
.sm*X*	Sets, in a smaller point size, the value specified by the *X* variable (macro).
.so	Indicates source input file (macro).
\n(so	Sets additional section title offset (number register).
.sp	Indicates vertical space (macro).
\n(sp	Indicates section title point size (number register).
\n(ss	Indicates section prespace (number register).
.sx	Changes section depth (macro).
.sz +*Number*	Augments point size by the specified number of points (macro).
.ta	Sets tab stops (macro).
.tc	Sets tab repetition character (macro).
***(td**	Sets today's date (string).

n(tf	Indicates title font (number register).
.th	Produces paper in thesis format. Must be used before initialization (macro).
.ti	Indicates temporary indent, next line only (macro).
.tl	Indicates 3-part title (macro).
\n(tm	Sets top title margin (number register).
.tp	Begins title page (macro).
\n(tp	Sets title point size (number register).
.tr	Translates (macro).
.u *X*	Underlines the value specified by the *X* variable, even in the **troff** command. No-fill mode only (macro).
.uh	Sets section head to follow; font automatically bold. Similar to the **.sh** macro, but unnumbered (macro).
.ul	Underlines next line (macro).
\v'*Distance***'**	Local vertical motion for the specified distance (**troff** command built-in function).
***v**	Inverts v for Czech e (string).
\w'*String***'**	Returns width of the specified string (**troff** command built-in function).
.xl	Sets local line length (macro).
.xp*Index*	Prints the specified index (macro).
\n(xs	Sets index entry prespace (number register).
\n(xu	Sets index indent, from right margin (number register).
\n(yr	Indicates year, last two digits only (number register).
\n(zs	Sets floating keep pre- or post-space (number register).
\{	Begins conditional group (**troff** command built-in function).
|	1/6 em, narrow space (**troff** command built-in function).
\}	Ends conditional group (**troff** command built-in function).
***~**	Indicates tilde (string).

For further information, see the *-ME Reference Manual* by E. P. Allman.

mm Macro Package for the mm, mmt, nroff, and troff Commands

The **mm** macro package provides macros to format text in a wide variety of document forms, such as memos, letters, and reports. The manner in which you type and edit a document is essentially independent of whether the document is later formatted at a terminal or phototypeset.

The **col** command may be required to postprocess **nroff** output. See the **col** command for specific requirements.

The **mm** macros and additional information are summarized under the following headings:

- Beginning Macros for Formal Memoranda
- Business Letter Macros
- Ending Macros (Trailing Information)
- Paragraphs
- Section Headings
- Lists
- Displays, Tables, Equations, and Footnotes
- Page Headers and Footers
- Miscellaneous Macros
- **mm** Registers
- **mm** Strings
- String Names
- Reserved Names.

Beginning Macros for Formal Memoranda

.ND *Date*
Sets new date.

.TL [*ChgNumber*] [*FileNumber*]
Sets title information. Text on the following line is used as the title of the document.

.AF [*CompanyName*]
Specifies author's company name.

.AU *Name* [*Initials*] [*Loc*] [*Dept*] [*Ext*] [*Room*] [*Option...*]
Sets author information.

.AT *AuthorTitle* [*...*]
Specifies title to follow signer's name (up to nine options).

.TM [*Number*]
Sets technical memorandum number.

.AS [**0** | **1** | **2**] [*Indent*]
Starts abstract, for technical memorandum and released paper only:

- **0** Abstract on cover sheet and first page
- **1** Abstract only on cover sheet
- **2** Abstract only on memorandum for file cover sheet.

.AE
Ends abstract.

.NS
Starts notation, allowed on memorandum for file cover sheets following an **.AS 2/.AE** macro pair (see "Ending Macros").

.NE
Ends notation, allowed on memorandum for file cover sheets following an **.AS 2/.AE** macro pair (see "Ending Macros").

.OK [*Keyword ...*]
Specifies other keywords (up to nine options).

.MT [*type*] [*title*]
Sets document type:

- **""** No type.
- **0** No type (internal letter).
- **1** Memorandum for file.
- **2** Programmer's notes.
- **3** Engineer's notes.
- **4** Released paper.
- **5** External letter.
- *"String"* The specified string is printed.

Title
User-supplied text prefixed to page number

Business Letter Macros

.WA
Starts writer's address.

.WE
Ends writer's address.

.LO CN [*Notation*]
Specifies confidential notation.

.LO RN [*Notation*]
Specifies reference notation.

.IA
Starts inside (recipient's) address.

.IE
Ends inside (recipient's) address.

.LO AT [*Notation*]
Specifies attention line.

.LO SA [*Notation*]
Specifies salutation.

.LO SJ [*Notation*]
Specifies subject line.

.LT [{ none **BL SB FB SP**}]

Specifies business letter type:

none	Blocked
BL	Blocked
SB	Semiblocked
FB	Full-Blocked
SP	Simplified.

Ending Macros (Trailing Information)

.FC [*Closing*]
Prints formal closing.

.SG [*Initials*] [**1**]
Prints signature line.

.NS [{" **"0 1 2 3 4 5 6 7 8 9 10 11 12 13** *String*}]
Starts notation:

" "

	Copy to
0	Copy to
1	Copy (with attachment) to
2	Copy (without attachment) to
3	Attachment
4	Attachments
5	Enclosure
6	Enclosures
7	Under Separate Cover
8	Letter to
9	Memorandum to
10	Copy (with attachments) to
11	Copy (without attachments) to
12	Abstract Only to
13	Complete Memorandum to
String	Copy (*String*) to.

.NE
Ends notation.

.AV *Name* [**1**]
Prints approval signature.

.CS [*Pgs*] [*Other*] [*Tot*] [*Figs*] [*Tbls*] [*Ref*]
Prints cover sheet.

.TX
Calls user exit for table-of-contents titles.

.TY
Calls user exit for table-of-contents header.

.TC [*Slev*] [*Spacing*] [*Tlev*] [*Tab*] [*H1*] [*H2*] [*H3*] [*H4*] [*H5*]
Prints table of contents.

Paragraphs

.P [{**0 1 2**}] Starts paragraph:

- **0** Left-justified (default)
- **1** Indented
- **2** Indented except after **.H**, **.LE**, **.DE**.

Section Headings

.H {**1 2 3 4 5 6 7**} [*HeadingText*] [*FootnoteMark*] Specifies numbered headings.

.HU *HeadingText* Specifies unnumbered headings.

.HM {**1 0001 A a I i**}**...** Specifies heading mark style:

- **1** Arabic
- **0001** Arabic with leading 0s (zeros)
- **A** Uppercase alphabetic
- **a** Lowercase alphabetic
- **I** Uppercase Roman
- **i** Lowercase Roman.

.HX [*Dlev*] [*Rlev*] [*HeadingText*] Calls user-defined exit macro before headings.

.HY [*Dlev*] [*Rlev*] [*HeadingText*] Calls user-defined exit macro in the middle of headings.

.HZ [*Dlev*] [*Rlev*] [*HeadingText*] Calls user-defined exit macro after headings.

Lists

If the last option [**1**] is present in the list-start macros, there is no space between items.

.AL [{**1 A a I i**}] [*TextIndent*] [**1**] Starts automatically incremented list (**1**).

.BL [*TextIndent*] [**1**] Starts a bullet list.

.DL [*TextIndent*] [**1**] Starts a dash list.

.ML *Mark* [*TextIndent*] [**1**] Starts a list in which each list item is tagged with a specified mark. If the value of the *TextIndent* is **NULL** or omitted, it is set to [*Mark* - width + 1]. If the 3rd argument is specified, no blank lines separate items in the list.

.RL [*TextIndent*] [**1**] Starts a reference list.

.VL *TextIndent* [*MarkIndent*] [**1**] Starts a variable tag list.

.LI [*Mark*] [**1**] Starts list item; **1** means that the *Mark* variable value is to be prefixed to the current mark.

.LE [**1**] Ends list item; **1** means to output a blank line after list. The default is no blank line.

.LB *TextIndent MarkIndent Pad Type* [*Mark*] [{**0 1**}] [{**0 1**}] Begins list:

The value of the *Type* variable is:

1=. 2=) 3=() 4=[] 5=<> 6={}.

Sixth option:

- **0** No blank line before each list item.

Seventh option:

- **0** No blank line before list.

.LC [*Level***]** Clears list status up to the *Level* variable value.

Displays, Tables, Equations, and Footnotes

.DS [{**0 1 2 3** }] [{**0 1**}] [*Number*]

.DS [{L I C CB}] [{N F}] [*Number***]**
Starts static display:

0 or L
No indent

1 or I Indent from left

2 or C
Center each line

3 or CB
Center as a block

0 or N
No-fill

1 or F
Fill.

Number
Indent from right the number of spaces specified by the *Number* parameter.

.DF [{**0 1 2 3** }] [{**0 1**}] [*Number*]

.DF [{L I C CB}] [{N F}] [*Number***]**
Starts floating display:

0 or L
No indent

1 or I Indent from left

2 or C
Center each line

3 or CB
Center as a block

0 or N
No-fill

1 or F
Fill.

Number
Indent from right the number of spaces specified by the *Number* parameter.

.DE Ends display.

.FG [*Title***] [***Override***] [0 1 2]**
The value of the *Override* variable replaces or enhances the default numbering. Specifies figure caption:

0 *Override* value is used as a prefix.

1 *Override* value becomes a suffix.

2 Replace *Override* value becomes a replacement.

.TS [H]
Starts table:

H Multipage table.

.TH [N]
Must be used when specifying option **H** to **.TS**:

N Suppresses table headers unless on top of new page.

.TE Ends table.

.TB [*Title***] [***Override***] [0 1 2]**
The value of the *Override* variable replaces or enhances the default numbering. Specifies table caption:

0 *Override* value is used as a prefix.

1 *Override* value becomes a suffix.

2 Replace *Override* value becomes a replacement.

.EX [*Title***] [***Override***] [0 1 2]**
The value of the *Override* variable replaces or enhances the default numbering. Specifies exhibit caption:

0 *Override* value is used as a prefix.

1 *Override* value becomes a suffix.

2 Replace *Override* value becomes a replacement.

.EQ [*Label***]**
Starts equation display using the specified label.

.EN Ends equation display.

.EC [*Title***] [***Override***] [0 1 2]**
The value of the *Override* variable replaces or enhances the default numbering. Specifies equation caption:

0 *Override* value is used as a prefix.

1 *Override* value becomes a suffix.

2 Replace *Override* value becomes a replacement.

.FS [*Label***]**
Starts footnote using the specified label as an indicator. Default is numbered footnote.

.FE Ends footnote.

.FD [{0 1 2 3 4 ... 11}] [1]
Sets footnote format:

First option:

Set up formatting style for footnote text. Default is 0 for **mmt** command. Default is 10 for **mm** command. See the following table for the value.

Second option:

Reset footnote counter on first-level heading.

.FD Arg.	Format
0	**Hyphens** **.nh** **Adjusted** **.ad** **Text Indented** Yes **Label Justified** Left
1	**Hyphens** **.hy** **Adjusted** **.ad** **Text Indented** Yes **Label Justified** Left
2	**Hyphens** **.nh** **Adjusted** **.na** **Text Indented** Yes **Label Justified** Left
3	**Hyphens** **.hy** **Adjusted** **.na** **Text Indented** Yes **Label Justified** Left
4	**Hyphens** **.nh** **Adjusted** **.ad** **Text Indented** No **Label Justified** Left

.FD Arg.	Format
5	**Hyphens** **.hy** **Adjusted** **.ad** **Text Indented** No **Label Justified** Left
6	**Hyphens** **.nh** **Adjusted** **.na** **Text Indented** No **Label Justified** Left
7	**Hyphens** **.hy** **Adjusted** **.na** **Text Indented** No **Label Justified** Left
8	**Hyphens** **.nh** **Adjusted** **.ad** **Text Indented** Yes **Label Justified** Right
9	**Hyphens** **.hy** **Adjusted** **.ad** **Text Indented** Yes **Label Justified** Right

.FD Arg.	Format
10	**Hyphens** **.nh** **Adjusted** **.na** **Text Indented** Yes **Label Justified** Right
11	**Hyphens** **.hy** **Adjusted** **.na** **Text Indented** Yes **Label Justified** Right

Page Headers and Footers

.PH "'*Left*'*Center*'*Right*'"	Specifies page header.
.OH "'*Left*'*Center*'*Right*'"	Specifies odd-page header.
.EH "'*Left*'*Center*'*Right*'"	Specifies even-page header.
.PF "'*Left*'*Center*'*Right*'"	Specifies page footer.
.OF "'*Left*'*Center*'*Right*'"	Specifies odd-page footer.
.EF "'*Left*'*Center*'*Right*'"	Specifies even-page footer.
.BS	Starts bottom-block.
.BE	Ends bottom-block.
.PX	Calls user exit for page-header.
.TP	Calls top of page macro.

Miscellaneous Macros

.B [*Option*] [*Prev-Font-option*]	Prints in bold (up to six options).
.I [*Option*] [*Prev-Font-option*]	Prints in italics (up to six options); underlines with the **nroff** command.
.R	Returns to Roman font.
.PM [*Option*]	Sets proprietary marking. If you do not give the **.PM** macro an option, you turn off proprietary markings. The **/usr/lib/macros/string.mm** file contains some proprietary markings. This file should be edited to meet the user's needs.
.RD [*Prompt*] [*Diversion*] [*String*]	Stops code macro. The *Prompt* variable should be a user-defined string without spaces. The *Diversion* variable allows the typed-in text to be saved. The *String* variable contains the first line typed following the prompt.

.RP [{**0 1** }] [{**0 1 2 3**}]
Produces reference page:

First option:

0 Resets reference counter (default).

1 Does not reset reference counter.

Second option:

0 Causes an **.SK** macro after (default).

1 Does not cause an **.SK** macro after.

2 Does not cause an **.SK** macro before.

3 Does not cause an **.SK** macro before or after.

.RS/.RF
Numbers references automatically.

.WC [{**N WF -WF FF -FF WD -WD FB -FB**}]
Controls width for footnotes and displays when using two columns:

N Normal mode (**-WF**, **-FF**, **-WD**).

WF Footnotes always wide.

-WF Footnotes follow page style.

FF First footnote determines width of remaining footnotes on that page.

-FF Footnotes follow setting of **WF** or **-WF** option.

WD Always wide displays.

-WD Displays follow page style.

FB Floating display causes page break (default).

-FB Floating display does not cause page break.

.SP [*Lines*]
Skips lines down.

.SK [*Number*]
Skips the specified number of pages. (The default is 1.)

.OP
Breaks to an odd page.

.2C
Prints output in two columns.

.1C
Prints output in one column (normal line width restored).

.SA [*Option*]
Sets right-margin justification

Options:

0 Sets default to **off** (default for the **nroff** command).

1 Sets default to **on** (default for the **troff** command).

If no option is specified, macro reverts to current default.

.SM *String1* [*String2*] [*String3*]
Reduces size of the *String1* variable value by 1 point if the *String3* variable value is omitted; otherwise, reduces size of the *String2* variable value by one point.

.HC *Character*
Sets hyphenation character to the *Character* variable value.

.S [*PointSize*] [*VerticalSpacing*] Sets point size and vertical spacing (the **troff** command only).

Defaults:

Point size = **10p**

Vertical spacing = **12p**

Options 1 and 2:

Number
New value.

+/-Number
Increment to current value.

D Default.

C Current value.

P Previous value.

.VM [*Top*] [*Bottom*] Sets variable vertical margins.

.nP Starts double-line indent on paragraph.

The following macros are for alternating fonts and all take one to six options:

.IB Alternates italics (underlines for **nroff**) and bold.
.BI Alternates bold and italics.
.RI Alternates Roman and italics.
.IR Alternates italics (underlines for **nroff**) and Roman.
.RB Alternates Roman and bold.
.BR Alternates bold and Roman.

mm Registers

If an * (asterisk) follows a register name, that register can be set one of two ways: from the command line (see the example in the **mm** command) or before the formatter reads **mm** macro definitions. In the following list, the number shown in parentheses is the default value.

A * Handle preprinted forms.
Au Inhibit author information on first page (1).
C * Copy type (such as Original and Draft) (0).
Cl Contents level (2).
Cp Placement of figures, tables, equations, and exhibits (1).
D * Debug flag (0). If set to 1, the **mm** command continues even if it encounters an error that is usually fatal.
De Eject page after floating displays (0).
Df If set to 1, format register for floating displays (5).
Ds Static display pre- and post-space (1).
E * Control font of the Subject/Date/From fields (0): 0 = bold; 1 = Roman.

0 Bold (0)

1 Roman.

Ec Equation counter.
Ej Page-ejection flag for headings (0).
Eq Equation label placement (0).
Ex Exhibit counter.
Fg Figure counter.
Fs Vertical footnote separation (1).
H1...H7 Heading counters.

Hb Heading break level (after **.H** and **.HU**) (2).
Hc Heading centering level for **.H** and **.HU** (0).
Hi Heading temporary indent (after **.H** and **.HU**) (1).
Hs Heading space level (after **.H** and **.HU**) (2).
Ht Heading type:

0 Concatenated numbers (0)

1 Single numbers (0).

Hu Heading level for unnumbered heading (2).
Hy Hyphenation control:

0 No hyphenation (0)

1 Enable hyphenation.

L * Length of page (66v).
Le List of equations following table of contents (0):

0 Do not print

1 Print.

Lf List of figures following table of contents (0):

0 Do not print

1 Print.

Li List indent (5, **troff** command); (6, **nroff** command).
Ls List level down to which there is spacing between items (6).
Lt List of tables following table of contents (0):

0 Do not print

1 Print

Lx List of exhibits following table of contents (1):

0 Do not print

1 Print.

N * Numbering style (0).
Np Numbered paragraphs:

0 Unnumbered

1 Numbered (0).

O * Offset of page.
Oc Page numbering style for table of contents:

0 Lowercase Roman

1 Arabic (0).

Of Figure caption style (0).
P Page number; managed by the **mm** command (0). The register accepts a value of 0, or positive integers.
Pi Paragraph indent (5).
Ps Paragraph spacing (1).
Pt Paragraph type (0).
Pv PRIVATE header:

0 Do not print PRIVATE

1 On first page only

2 On all pages (0).
Rf Reference counter; used by **.RS** macro.
S * The **troff** command's default point size (10).

Si Display indent (5).
T * Type of the **nroff** command output device (0).
Tb Table counter.
U * Underlining style (the **nroff** command) for **.H** and **.HU** (0).
W * Width of page (line and title length).

mm Strings

Print special strings by using the following escape sequences:

***x** For strings with single-character names (**x**)
***(xx** For strings with two-character names (**xx**).

String Names

BU Bullet.
Ci Indent of heading levels in the table of contents.
DT Current date. The locale-specific date format specified by the locale setting for the **LC_TIME** category is used as the default setting. This corresponds to the **%x** format specifier of the **strftime** subroutine. Use the **.ND** macro to change the current date.
EM Em dash.
F Footnote numbering.
HF Heading level font string:

- **1** Roman
- **2** italics
- **3** Bold (2 2 2 2 2 2 2).

HP Point sizes of the various heading levels.
Le Title of the list of equations.
Lf Title of the list of figures.
Lt Title of the list of tables.
Lx Title of the list of exhibits.
RE SCCS SID of **mm** macros.
Rf Reference numberer.
Rp Title of the reference page.
Tm Trademark.
` Grave accent.
' Acute accent.
^ Circumflex.
~ Tilde.
: Lowercase umlaut.
; Uppercase umlaut.
, Cedilla.

Reserved Names

If you define your own strings, macros, and registers, use only names that consist of either a single lowercase letter, or a lowercase letter followed by any character other than a lowercase letter. The names **c2** and **nP** are exceptions to this; they are reserved.

mptx Macro Package for the nroff and troff Commands

The **mptx** macro package provides a definition for the **.xx** macro that is used for formatting a permuted index produced by the **ptx** command. The **mptx** macro package does not provide any other formatting

capabilities, such as headers and footers. Use the **mptx** macro package in conjunction with the **mm** macro package if such capabilities are required. In this case, call the **-mptx** option after the **-mm** call, as follows:

nroff -mm -mptx *File...* | *Printer*

ms Macro Package for the nroff and troff Commands

The ms macro package of **nroff** and **troff** command macro definitions provides a formatting facility for various styles of articles, theses, and books. In certain cases, the **col** command may be required to postprocess output.

The macro requests are defined in the **ms** Requests section. Many **nroff** and **troff** command requests can have unpredictable results in conjunction with this package. However, the first 4 requests in the following list can be used after initialization, and the last 2 requests can be used before initialization.

.bp	Begins new page.
.br	Breaks output line.
.ce [*Number*]	Centers the next specified number of lines.
.ls [*Number*]	Sets line spacing. Set the value of the *Number* variable to 1 (one) to single-space text; and to 2 to double-space text.
.na	Turns off alignment of right margin.
.sp [*Number*]	Inserts the specified number of spacing lines.

Font and point-size changes with the**\f** and **\s** macros are also allowed. For example, `\fIword\fR` italicizes `word`. Output of the **tbl**, **eqn**, and **refer** command preprocessors for equations, tables, and references is acceptable as input.

Formatting distances can be controlled in **ms** macros by means of built-in number registers. For example, the following number register sets the line length to 6.5 inches:

```
.nr LL 6.5i
```

For more information on **ms** macro registers, see **ms** Registers.

ms Requests

Following are external **ms** macro requests:

.AB [**X**]	Begins abstract. If **X** is no, do not label abstract. Initial Value: **-** Break: **yes**
.AE	Ends abstract. Break: **yes**Initial Value: **-** Break: **yes**
.AI*Name*	Author's institution. Initial Value: **-** Break: **yes**
.AM	Sets accent mark definitions. Initial Value: **-** Break: **no**

.AU*Name* Sets author's name.

Initial Value: **-**

Break: **yes**

.B [**X**] Puts **X** in boldface. If no **X**, switches to boldface.

Initial Value**: -**

Break: **no**

.B1 Begins text to be enclosed in a box.

Initial Value: **-**

Break: **yes**

.B2 Ends boxed text and prints it.

Initial Value: **-**

Break: **yes**

.BT Prints bottom title at foot of page.

Initial Value: **date**

Break: **no**

.BX X Prints word **X** in a box.

Initial Value: **-**

Break: **no**

.CM Cuts mark between pages.

Initial Value: **if t**

Break: **no**

.CT Indicates chapter title; page number moved to CF (TM).

Initial Value: **-**

Break: **yes**

Reset: **yes**

.DA [**X**] Forces date **X** at bottom of page. If no **X**, date is today.

Initial Value: **if n**

Break: **no**

.DE Ends display (unfilled text) of any kind.

Initial Value: **-**

Break: **yes**

.DS X Y Begins display with keep. **X**=I, L, C, B; **Y**=indent.

Initial Value: **I**

Break: **yes**

.ID Y Indents display with no keep; **Y**=indent.

Initial Value: **8n, .5i**

Break: **yes**

.LD Sets left display with no keep.

Initial Value: **-**

Break: **yes**

.CD Centers display with no keep.

Initial Value: **-**

Break: **yes**

.BD Block display; centers entire block.

Initial Value: **-**

Break: **yes**

.EF X Sets even page footer **X** (3 part as for **troff** command, **.tl** request).

Initial Value: **-**

Break: **no**

.EH X Sets even page header **X** (3 part as for **troff** command, **.tl** request).

Initial Value: **-**

Break: **no**

.EN Ends displayed equation produced by **eqn** command.

Initial Value: **-**

Break: **yes**

.EQ [**X**] [**Y**] Breaks out equation. **X**=L, I, C; **Y** is equation number.

Initial Value: **-**

Break: **yes**

.FE Ends footnote to be placed at bottom of page.

Initial Value: **-**

Break**: no**

.FP Numbers footnote paragraph; can be redefined.

Initial Value: **-**

Break: **no**

FS [**X**] Starts footnote; **X** is optional footnote label.

Initial Value: **-**

Break: **no**

.HD Sets optional page header below header margin.

Initial Value: **undef**

Break: **no**

.I [**X**] Italicizes **X**. If no **X**, equivalent to italics font **.ft 2**.

Initial Value: **-**

Break: **no**

.IP X Y Indents paragraph, with hanging tag **X**. **Y** specifies spaces to indent.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.IX X Y Indexes words such as **X** and **Y**, up to five levels.

Initial Value: **-**

Break: **yes**

.KE Ends keep of any kind.

Initial Value: **-**

Break: **no**

.KF Begins floating keep; text fills remainder.

Initial Value: **-**

Break: **no**

.KS Begins keep; keeps unit together on a single page.

Initial Value: **-**

Break: **yes**

.LG Sets larger type size; increases point size by 2. Valid only for the **troff** command.

Initial Value: **-**

Break: **no**

.LP Begins left block paragraph.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.MC X Sets multiple columns. **X** is column width.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.ND [**X**] Indicates no date in page footer; **X** is date on cover.

Initial Value: **if t**

Break: **no**

.NH X Y Sets numbered header: **X**=level; **X**=0, resets; **X**=S, sets to **Y**.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.NL Sets point size back to default. Valid for the **troff** command only.

Initial Value: **10p**

Break: **no**

.OF X Sets odd page footer **X** (3 part as for **me** macro, **.tl** request).

Initial Value: **-**

Break: **no**

.OH X Sets odd page header **X** (3 part as for **me** macro, **.tl** request).

Initial Value: **-**

Break: **no**

.P1 Prints header on first page.

Initial Value: **if TM**

Break: **no**

.PP Indents first line of paragraph.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.PT Prints page title at head of page.

Initial Value: **%**

Break: **no**

.PX X Prints index (table of contents); **X**=do not suppress title.

Initial Value: **-**

Break: **yes**

.QP Quotes paragraph (indented and shorter).

Initial Value: **-**

Break: **yes**

Reset: **yes**

.R [**X**] Returns to Roman font. Prints in Roman font. If **X** is missing, equivalent to font **.ft1**.

Initial Value: **on**

Break**: no**

.RE Retreats (end level of relative indentation). Used with the **.RS** request.

Initial Value: **5n**

Break: **yes**

Reset: **yes**

.RP [**X**] Prints title page in released paper format; **X**=no, stops title on first page.

Initial Value: **-**

Break: **no**

.RS Right-shifts in one indentation level (start level of relative indentation). Used with the **.IP** request.

Initial Value: **5n**

Break: **yes**

Reset: **yes**

.SG Sets signature line.

.SH Sets unnumbered section header (in boldface).

Initial Value: **-**

Break: **yes**

Reset: **yes**

.SM Sets smaller type size; decrease point size by 2. Valid for the **troff** command only.

Initial Value: **-**

Break: **no**

.TA Sets tabs to 8n, 16n, ... (**nroff**); 5n, 10n, ... (**troff**).

Initial Value: **8n**, **5n**

Break: **no**

.TC X Prints table of contents at end; **X**=do not suppress title.

Initial Value: **-**

Break: **yes**

.TE Ends table processed by **tbl** command.

Initial Value: **-**

Break: **yes**

.TH Ends multipage header of table. Must be used with the **.TS** H request.

Initial Value: **-**

Break: **yes**

.TL Sets title line (in boldface and 2 points larger).

Initial Value: **-**

Break: **yes**

.TM Sets UC Berkeley thesis mode.

Initial Value: **off**

Break: **no**

.TS X Begins table. If **X** is H, table prints header on all pages.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.UL X Underlines **X**, even for the **troff** command.

Initial Value: **-**

Break: **no**

.UX X Sets UNIX; trademark message first time; **X** appended.

Initial Value: **-**

Break: **no**

.XA X Y Sets another index entry; **X**=page; **X**=no, for none.

Initial Value: **-**

Break: **yes**

.XE Ends index entry or series of **.IX** request entries.

Initial Value: **-**

Break: **yes**

.XP Exdents first line of paragraph; others indented.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.XS X Y Begins index entry; **X**=page; **X**=no, for none; **Y**=indent.

Initial Value: **-**

Break: **yes**

.1C Begins one-column format, on a new page.

Initial Value: **on**

Break: **yes**

Reset: **yes**

.2C Begins two-column format.

Initial Value: **-**

Break: **yes**

Reset: **yes**

.]- Sets beginning of **refer** command reference.

Initial Value: **-**

Break: **no**

.[0 Sets end of unclassifiable type of reference.

Initial Value: **-**

Break: **no**

.[N For journal article, **N**=1 (one). For book, **N**=2. For book article, **N**=3.

Initial Value: **-**

Break: **no**

ms Registers

Following is a list of number registers and their default values:

PS Sets point size. Takes effect for paragraph. Default is 10.
VS Sets vertical spacing. Takes effect for paragraph. Default is 12.
LL Sets line length. Takes effect for paragraph. Default is 6i.

LT Sets title length. Takes effect on next page. Defaults to the **LL** register value.
FL Sets footnote length. Takes effect at next **.FS** request. Default is 5.5i.
PD Sets paragraph distance. Takes effect for paragraph. Default is 1v (in **nroff**), .3v (in **troff**).
DD Sets display distance. Takes effect for displays. Default is 1v (in **nroff**), .5v (in **troff**).
PI Sets paragraph indent. Takes effect for paragraph. Default is 5n.
QI Sets quotation indent. Takes effect at next **.QP** request. Default is 5n.
FI Sets footnote indent. Takes effect at next **.FS** request. Default is 2n.
PO Sets page offset. Takes effect on next page. Default is 0 (zero) (in **nroff**), 1i (in **troff**).
HM Sets header margin. Takes effect on next page. Default is 1i.
FM Sets footer margin. Takes effect on next page. Default is 1i.
FF Sets footnote format. Takes effect at next **.FS** request. Default is 0 (zero) (1, 2, 3 available).

When resetting number register values, make sure to specify the appropriate units. Set the line length to 7i instead of just 7, which would result in output with one character per line. Setting the **FF** register to 1 (one) suppresses footnote superscripting. Setting it to 2 also suppresses indentation of the first line. Setting the **FF** register to 3 produces a footnote paragraph like the .**IP** request.

Following is a list of string registers available in the **ms** macros. These string registers can be used anywhere in the text.

***Q** Open quotation marks (**”** in **nroff**; ` ` in **troff**)
***U** Close quotation marks (**“** in **nroff**; ’ ’ in**troff**)
***-** Dash (**—** in **nroff**; **-** in **troff**)
***(MO** Month of year
***(DY** Day (current date)
****** Automatically numbered footnote
***’** Acute accent (before letter)
***`** Grave accent (before letter)
***^** Circumflex accent (before letter)
***,** Cedilla (before letter)
***:** Umlaut (before letter)
***~** Tilde (before letter).

When using the extended accent mark definitions available with the **.AM** request, these strings should come after, rather than before, the letter to be accented.

Notes:

1. It is important to note that floating keeps and regular keeps are diverted to the same space, so they cannot be mixed.
2. The date format is restricted to U.S. English format.

mv Macro Package for the mvt and troff Commands

This package simplifies the typesetting of view graphs and projection slides in a variety of sizes. Although a few macros accomplish most of the formatting tasks needed in making transparencies, the entire facilities of the **troff**, **tbl**, **pic**, and **grap** commands are available for more difficult tasks.

The output can be previewed on most terminals, in particular the Tektronix 4014. For this device, specify the **-rX1** flag (which is automatically specified by the **mvt** command when that command is called with the **-D4014** flag). To preview output on other terminals, specify the **-a** flag.

The **mv** macros are summarized under the following headings:

- Foil-Start Macros
- Level Macros

- Text-Control Macros
- Default-Setting Macros.

Foil-Start Macros

For the following nine macros, the first character of the name (**V** or **S**) distinguishes between view graphs and slides, respectively, while the second character indicates whether the foil is square (**S**), small wide (**w**), small high (**h**), big wide (**W**), or big high (**H**). Slides are narrower than the corresponding view graphs. The ratio of the longer dimension to the shorter one is larger for slides than for view graphs. As a result, slide foils can be used for view graphs, but view graphs cannot be used for slide foils. On the other hand, view graphs can accommodate a bit more text.

.VS [*FoilNumber*] [*FoilID*] [*Date*]
: Starts a square view graph. Foil size is to be 7 inches by 7 inches. The foil-start macro resets all variables (such as indent and point size) to initial default values, except for the values of the *FoilID* and *Date* variables inherited from a previous foil-start macro. The **.VS**macro also calls the **.A** macro.

.Vw, **.Vh**,**.VW**, **.VH**, **.Sw**, **.Sh**, **.SW**, **.SH**
: Same as the **.VS** macro, except that these macros start view graphs (**V**) or slides (**S**) that are small wide (**w**), small high (**h**), large wide (**W**), or large high (**H**).

  The following macros are recommended:

  - **.VS** for square view graphs and slides
  - **.Sw** (and, if necessary, **.Sh**) for 35mm slides.

.Vw [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 5 inches high.

.Vh [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 5 inches wide by 7 inches high.

.VW [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 5.4 inches high.

.VH [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 9 inches high.

.Sw [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 5 inches high.

.Sh [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 5 inches wide by 7 inches high.

.SW [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 5.4 inches high.

.SH [*FoilNumber*] [*FoilID*] [*Date*]
: Same as the **.VS** macro, except that foil size is 7 inches wide by 9 inches high.

Note: The **.VW** and **.SW** foils are meant to be 9 inches wide by 7 inches high. However, because the typesetter paper is generally only 8 inches wide, **.VW** and **.SW** foils are printed 7 inches wide by 5.4 inches high and have to be enlarged by a factor of 9/7 before use as view graphs.

Level Macros

.A [*X*]
: Places text that follows at the first indentation level (left margin). The presence of the *X* variable suppresses the half-line spacing from the preceding text.

.B [*Mark* [*Size*]]
: Places text that follows at the second indentation level. Text is preceded by a specified mark (default is a large bullet). The *Size* variable is the increment or decrement to the point size of the mark with respect to the *prevailing* point size (default is 0). A value of 100 for the *Size* variable makes the point size of the mark equal to the default value of the *Mark* variable.

.C [*Mark* [*Size*]]
: Same as the **.B** macro, but for the third indentation level. The default value of the *Mark* variable is an em dash.

.D [*Mark* [*Size*]]
: Same as the **.B** macro, but for the fourth indentation level. The default value of the *Mark* variable is a small bullet.

Text-Control Macros

.I [+/-] [*Indentation*] [A[X]] Changes the current text indent (does not affect titles). The specified indentation is in inches unless dimensioned. The default is 0. If the *Indentation* variable is signed, it is an increment or decrement. The presence of the *A* variable calls the **.A** macro and passes the *X* variable (if any) to it.

.S [*Size*] [*Length*] Sets the point size and the line length. The value specified in the *Size* variable is the point size (default is previous). If the *Size* variable value is 100, the point size reverts to the *initial* default for the current foil-start macro. If the *Size* variable is signed, it is an increment or decrement (default is 18 for the **.VS**, **.VH**, and **.SH** macros, and 14 for the other foil-start macros). The *Length*variable specifies the line length (in inches unless dimensioned; the default is 4.2 inches for the **.Vh** macro, 3.8 inches for the **.Sh** macro, 5 inches for the **.SH** macro, and 6 inches for the other foil-start macros).

.T *String* Prints the *String* variable value as a centered, enlarged title.

.U *String1*[*String2*] Underlines the *String1* variable value and concatenates the *String2 variable* value (if any) to it. Using this operation is not recommended.

Default-Setting Macros

.DF [*Number Name*]**...** Sets font positions. It cannot be displayed within foil input text; that is, it must follow the input text for a foil, but it must precede the next foil-start macro. The specified number is the position of the font specified by the *Name* variable. The **.DF** macro takes up to four pairs of *Number Name* variables, such as `1 H`. The first *Name* variable specifies the prevailing font. For example: **.DF 1 H 2 I 3 B 4 S**.

.DV [*A*] [*B*] [*C*] [*D*] Alters the vertical spacing between indentation levels. The value specified by the *A*, *B*, *C*, or *D* variable is the spacing for the **.A**, **.B**, **.C**, or **.D** macro, respectively. All non-null parameters must be dimensioned. Null parameters leave the corresponding spacing unaffected. The default setting is: **.DV .5v .5v .5v 0v**.

The **.S**, **.DF**, **.DV**, and **.U** macros do not cause a break. The **.I** macro causes a break only if it is called with more than one variable. All the other macros cause a break.

The **mv** macro package also recognizes the following uppercase synonyms for the following corresponding lowercase **troff** command requests:

- **.AD**
- **.BR**
- **.CE**
- **.FI**
- **.HY**
- **.NA**
- **.NF**
- **.NH**
- **.NX**
- **.SO**
- **.SP**
- **.TA**
- **.TI**

The **Tm** string produces the trademark symbol.

Environment Variable

LANG Determines the locale's equivalent of y for yes or no queries. The allowed affirmative responses are defined in the locale variable **YESSTR**. If **LANG** is not set, or if it is set to an empty string, the **YESSTR** from the default C locale is used.

nroff and troff Requests for the nroff and troff Commands

The following **nroff** and **troff** requests are included in a specified working file or in standard input. The **nroff** and **troff** requests control the characteristics of the formatted output when the file or standard input is processed with the **nroff** or **troff** commands. The **nroff** and **troff** requests are grouped by function, in the following sections:

- Numerical Parameter Input
- Font and Character Size Control
- Page Control
- Text Filling, Adjusting, and Centering
- Vertical Spacing
- Line Length and Indenting
- Macros, Strings, Diversions, and Position Traps
- Number Registers
- Tabs, Leaders, and Fields
- Input and Output Conventions and Character Translations
- Hyphenation
- Three-Part Titles
- Output Line Numbering
- Conditional Acceptance of Input
- Environment Switching
- Insertions from Standard Input
- Input and Output File Switching
- Miscellaneous

For number variables written as *+Number*, the variable can be expressed as follows:

- The *Number* variable by itself is an absolute value.
- The *+Number* variable increases the currently set value.
- The *-Number* variable decreases the variable relative to its current value.

Note: For all numeric parameters, numbers are expressed using ASCII Arabic numerals only.

The notes at the end of this command are referenced in the specific requests where applicable.

Numerical Parameter Input

Both **nroff** and **troff** requests accept numerical input with the appended scale indicators shown in the following table, where *S* is the current type size in points, *V* is the current vertical line spacing in basic units, and *C* is a nominal character width in basic units.

Scale		**Number of Basic nroff Units**
Indicator	**Meaning**	
i	Inch (machine-dependent for troff)	240
c	Centimeter	240x50/127

Scale		Number of Basic nroff Units
Indicator	**Meaning**	
P	Pica = 1/6 inch	240/6
m	Em = S points	*C*
n	En = Em/2	*C (same as Em)*
p	Point = 1/72 inch	240/72
u	Basic unit	1
v	Vertical line space	*V*
k	Width single-width kana	*C*
K	Width double-width kanji	Two *C*s
none	Default	

Note:

1. If a non-kanji output device is selected, an en-width is used instead.
2. If a non-kanji output device is selected, an em-width is used instead.

In the **nroff** request, both the em and the en are taken to be equal to the *C*, which is output-device dependent; frequent values are 1/10 and 1/12 inch. Actual character widths in the **nroff** request need not be all the same, and characters constructed with predefined strings such as **- >** are often extra wide.

Japanese Language Support: In the output from the **nroff** command, all double-width Japanese characters such as all kanji and some katakana characters have a fixed width equal to two *C*s. All single-width Japanese characters such as some katakana characters have a fixed width equal to *C*.

The scaling for horizontally-oriented control characters, vertically-oriented control characters, and the requests **.nr**, **.if**, and **.ie** are as follows:

Orientation	**Default Measure**	**Request or Function**
Horizontal	Em (**m**)	**.ll, .in, .ta, .lt, .po, .mc, \h, \l**
Vertical	Vertical line space (**v**)	**.pl, .wh, .ch, .dt, .sp, .sv, .ne, .rt, \v \x, \L**
Register-oriented or Conditional	Basic unit (**u**)	**.nr, .if. .ie**
Miscellaneous	Point (**p**)	**.ps, .vs, \H, \s**

All other requests ignore scale indicators. When a number register containing an already appropriately scaled number is interpreted to provide numerical input, the unit scale indicator **u** might need to be appended to prevent an additional inappropriate default scaling. The *Number* might be specified in decimal-fraction form, but the parameter that is finally stored is rounded to an integer number of basic units.

Font and Character Size Control

.bd *Font Number*

Makes the characters in the specified font artificially bold by overstriking them the specified number of times when using **nroff**, or by printing each character twice separated by *Number* -1 basic units when using **troff**. If the *Number* variable is not specified, the bold mode is turned off. The *Font* value must be an ASCII font name or font position. For the **nroff** command, the default setting of the **.bd** request is 3 3, specifying that characters on the font mounted at position 3 (usually bold) are to be overstruck 3 times (that is, printed in place a total of 4 times).

The font name itself can be substituted for the font position; for example, **.bd I 3**. The *Number* variable is functionally identical to the **-u** flag of the **nroff** command. (The bold mode must be in effect when the characters are physically printed.) This request can affect the contents of the **.b** general-number register.

The bold mode still must be in effect, or restarted at the time of physical output. You cannot turn off the bold mode in the **nroff** command if it is being controlled locally by the printing device as with, for example, a DASI 300.

Initial Value: Off

If No Value Specified: **-**

.bd S *Font Number*

Makes the characters in the special font bold whenever the specified font is the current font. The mode must be in effect when the characters are physically printed. The *Font* value must be an ASCII font name or font position. The mode still must be in effect, or again so, at the time of physical output.

Initial Value: Off

If No Value Specified: **-**

.cs *Font Number M*

Sets constant character space (width) mode to the *Font* variable value (if mounted). The width of every character is taken to be the value specified in the *Number* variable divided by 36 ems. If the *M* variable is not specified, the em width is that of the character's point size; if the *M* variable is given, the width is the value specified by the *M* variable minus points. All affected characters are centered in this space, including those with an actual width larger than this space. Special font characters occurring while the specified font is the current font are also so treated. The *Font* value must be an ASCII font name or font position. If the *Number* variable is absent, the mode is turned off. The mode must be in effect when the characters are physically printed. This request is ignored by the **nroff** command. Relevant values are part of the current environment. The mode still must be in effect, or again so, at the time of physical output.

Initial Value: Off

If No Value Specified: **-**

.fp *Font Number*[*File*]

Specifies the font position. This is a statement that the specified font is mounted on the position specified by the *Number* variable. The *Font* variable must be a one- or two-character ASCII font name.

Attention: It is an irrecoverable error if the *Font* variable is not specified.

The **.fp** request accepts a third optional variable, the *File* variable, which is the actual path name of the file containing the specified font. The *File* variable value can be any legal file name and can contain extended characters.

Japanese Language Support**:** The *File*value can be any legal file name. Values are typesetter- or printer-dependent.

Initial Value: **-**

If No Value Specified: Ignored

.ft *Font*

Changes the font style to the specified font, or if *Font* value is numeric, to the font mounted on that position. Alternatively, embed **\f***Font* command. The font name **P** is reserved to mean the previous font. The *Font* variable value must be an ASCII font name or font position.

If using a font name consisting of two characters, use the alternative form of **.ft**, **\f**. Relevant values are part of the current environment. Values are typesetter or printer-dependent.

Initial Value: Roman

If No Value Specified: Previous

.ps [+/-][*Number*]

Sets the point size to that specified by the +/-*Number* variable. Although any positive size value can be requested, an invalid size results in the nearest valid size being used. Size 0 refers to the previous size. Alternatively, **\s***Number* or **\s**+/-*Number*; if the *Number* value is two digits, use **\s(***Number* or **\s**+/-**(***Number*. For compatibility with older versions of the **troff** command, the form is valid for two-digit values of *n* = **10**, **11**, **12**, **14**, **16**, **18**, **20**, **22**, **24**, **28**, and **36**.

This request is ignored by the **nroff** command. Relevant values are part of the current environment.

Initial Value: 10 point

If No Value Specified: Previous

.ss *Number*

Sets space-character size to the specified number divided by 36 ems. This size is the minimum word spacing in adjusted text. This request is ignored by the **nroff** command. Relevant values are part of the current environment.

Initial Value: 12/36 em

If No Value Specified: Ignored

Page Control

.bp [+/-][*Number*]

Specifies a break page. The current page is ejected and a new page is begun. If the +/-*Number* variable is specified, its value becomes the new page number. Also refer to the **.ns** request.

This request usually causes a line break similar to the **.br** request. Calling this request with the control character " **'** " (instead of "**.**") suppresses that break function.

Initial Value: *Number*=**1**

If No Value Specified: **-**

.mk *Register*

Marks the current vertical place (or a place in the current diversion) in an internal register (associated with the current diversion level) or in the specified register, if given. The *Register* variable is the ASCII name of a number register. Mode or relevant values are associated with the current diversion level. For more information, refer to the **.rt** request.

Initial Value: None

If No Value Specified: Internal

.ne *Number D*

Indicates a need for the specified vertical space. If the page space needed (*Number*) is greater than the distance to the next trap (*D*), a forward vertical space of size *D* occurs, which springs the trap. If there are no remaining traps on the page, the size specified by the *D* variable is the distance to the bottom of the page. If the distance to the next trap (*D*) is less than one vertical line space (**v**), another line could still be output before the trap is sprung. In a diversion, the size specified by *D* is the distance to the diversion trap, if any, or is very large.

The value of *D* is also usually contained in the **.t** *Number* register. Mode or relevant values are associated with the current diversion level.

Initial Value: *Number*=**1***V*

If No Value Specified: **-**

.pl [+/-][*Number*]

Sets page length to the +/-*Number* variable value. The internal limitation is approximately 136 inches in the **nroff** command, but varies with the device type in the **troff** command. A good working maximum for the **troff** command is 75 inches. The current page length is available in the **.p** register.

Initial Value: 11 inches

If No Value Specified: 11 inches

.pn [+/-][*Number*]

Specifies that the next page (when it occurs) has the page number specified by the +/-*Number* variable. A **.pn** request must occur either before text is initially printed or before a break occurs to affect the page number of the first page. The current page number is in the **%** register.

Initial Value: *Number*=**1**

If No Value Specified: Ignored

.po [+/-][*Number*]

Specifies a page offset. The current left margin is set to the +/-*Number*variable value. The initial **troff** command value provides 1 inch of left margin. For more information, refer to ”Line Length and Indenting“. The current page offset is available in the **.o** register.

Initial Value: **0** for the **nroff** command; **1** for the **troff** command.

If No Value Specified: Previous

.rt [+/-][*Number*]

Returns upward only to a marked vertical place in the current diversion. If the +/-*Number* variable value (relative to the current place) is given, the place is the value specified by the +/-*Number* variable from the top of the page or diversion. If the *Number* variable is not specified, the place is marked by a previous **.mk** request. Mode or relevant values are associated with the current diversion level.

The **.sp** request can be used in all cases, instead of the **.rt** request, by spacing to the absolute place stored in an explicit register as, for example, when using the sequence **.mk** *Register* . . . **.sp****n**R**u**.

Initial Value: None

If No Value Specified: Internal

Text Filling, Adjusting, and Centering

.ad *Indicator*

Begins line adjustment. If the fill mode is not on, adjustment is deferred until the fill mode is back on. If the *Indicator* variable is present, the adjustment type is changed as shown in the following list:

Indicator	Adjustment Type
l	Adjust left margin only.
r	Adjust right margin only.
c	Center.
b* or *n	Adjust both margins.
blank	Unchanged.

The adjustment indicator can also be a number obtained from the **.j** register.

Japanese Language Support:

Indicator	Adjustment Type
k	Turn on kinsoku shori processing (turned off with **.ad n**, **.ad b**, or **.ad l**).

Usually, lines of Japanese text are filled to the margins without regard for the characters beginning or ending lines. When kinsoku shori processing is enabled, lines are prevented from ending with an open bracket character or from beginning with a close bracket or punctuation character. If a line ends with an open bracket, the line is left short and the bracket begins the next line. If a line begins with a close bracket or punctuation character, the preceding line is extended and the character ends the preceding line. Requesting Japanese kinsoku shori processing on an output device that does not support kanji characters has no effect.

Relevant values are part of the current environment.

Initial Value: Adjust, both

If No Value Specified: Adjust

.br

Specifies a break. The filling of the line currently being collected is stopped and the line is output without adjustment. Text lines beginning with space characters and empty text lines (blank lines) also cause a break.

Initial Value: **-**

If No Value Specified: **-**

.ce [*Number*]

Centers the next specified number of input text lines within the current line length, minus indent. If the *Number* variable equals 0, any residual count is cleared. A break occurs after each of the *Number* variable input lines. If the input line is too long, it is left adjusted. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ’ “ (instead of ”**.**“) suppresses that break function.

Initial Value: Off

If No Value Specified: *Number*=**1**

.fi Fills subsequent output lines. The **.u** register has a value of 1 (one) in fill mode and a value of 0 (zero) in no-fill mode. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ’ “ (instead of ”**.**“) suppresses that break function.

Initial Value: Fill

If No Value Specified: **-**

.na Specifies no-adjust mode. Adjustment is turned off; the right margin is ragged. The adjustment type for the **.ad** request is not changed. Output-line filling still occurs if the fill mode is on. Relevant values are part of the current environment.

Initial Value: None

If No Value Specified: **-**

.nf Specifies no-fill mode. Subsequent output lines are neither filled nor adjusted. Input-text lines are copied directly to output lines without regard for the current line length. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ’ “ (instead of ”**.**“) suppresses that break function.

Initial Value: Fill

If No Value Specified: **-**

Vertical Spacing

Blank text line Causes a break and outputs a blank line exactly like an **.sp 1** request.

.ls *Number* Sets line spacing to the value specified by the +/-*Number* variable. The *Number* **-1** *V***s** (blank lines) variable values are appended to each output-text line. Appended blank lines are omitted if the text or previous appended blank line reached a trap position. Relevant values are part of the current environment.

Initial Value: **1**

If No Value Specified: Previous

.ns Turns on no-space mode. When on, the no-space mode inhibits **.sp** and **.bp** requests without a next page number. The no-space mode is turned off when a line of output occurs or with the **.rs** request. This request usually causes a break.

Initial Value: Space

If No Value Specified: **-**

.os Outputs saved vertical space. The no-space mode has no effect. Used to output a block of vertical space requested by the previous **.sv** request.

Initial Value: **-**

If No Value Specified: **-**

.rs Restores spacing. The no-space mode is turned off. This request usually causes a break.

Initial Value: None

If No Value Specified: **-**

.sp *Number*
Spaces vertically in either direction. If the *Number* variable value is negative, the motion is backward (upward) and is limited to the distance to the top of the page. Forward (downward) motion is truncated to the distance to the nearest trap. If the no-space mode is on, no spacing occurs. Refer to the **.ns** and **.rs** requests. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ”“ (instead of ”.“) suppresses that break function.

Initial Value: **-**

If No Value Specified: **1***V*

.sv *Number*
Saves a contiguous vertical block of the specified size. If the distance to the next trap is greater than the *Number* variable value, the specified vertical space is output. The no-space mode has no effect. If this distance is less than the specified vertical space, no vertical space is immediately output, but is remembered for later output (refer to the **.os** request). Subsequent **.sv** requests overwrite any still-remembered *Number* variable value.

Initial Value: **-**

If No Value Specified: *Number*=**1***V*

.vs *Number*
Sets vertical base-line spacing size *V* to the *Number* variable. Transient extra vertical space can be specified by **\x** *N*. Relevant values are part of the current environment.

Initial Value: The *Number* variable equals 1/16 inch for the **nroff** command and 12 points for the **troff** command.

If No Value Specified: Previous

Line Length and Indenting

.in [+/-]*Number*
Sets indent to the +/-*Number* variable value. The indent is prepended to each output line. Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ’ “ (instead of ”**.**“) suppresses that break function.

Initial Value: *Number*=**0**

If No Value Specified: Previous

.ll [+/-]*Number*
Sets line length to the +/-*Number* variable value. In the **troff** command, the maximum line length plus page offset is device-dependent. Relevant values are part of the current environment.

Initial Value: 6.5 inches

If No Value Specified: Previous

.ti [+/-]*Number*
Specifies a temporary indent. The next output text line is indented a distance of the value specified by the +/-*Number* variable with respect to the current indent. A negative value for the *Number* variable can result in spacing backward over the current indent, so that the resulting total indent can be a value of 0 (zero) (equal to current page offset), but cannot be less than the current page offset. The temporary indent applies only for the one output line following the request; the value of the current indent, which is stored in the **.i** register, is not changed.

Relevant values are part of the current environment. This request usually causes a line break similar to the **.br** request. Calling this request with the control character ” ’ “ (instead of ”**.**“) suppresses that break function.

Initial Value: **-**

If No Value Specified: Ignored

Macros, Strings, Diversions, and Position Traps

.am *Macro1* [*Macro2*]
Appends to *Macro 1*; appends version of the **.de** request. Both the *Macro1* and *Macro2* variables must be either one or two ASCII characters. *Macro2* is a termination sequence to end the diversion.

Initial Value: **-**

If No Value Specified: **.***Macro2***=..**

.as *StringName String*
Appends the specified string to the value specified by the *StringName* variable; appended version of the **.ds** request. The *StringName* variable value must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: Ignored

.ch *Macro* [*Number*]
Changes the trap position for the specified macro to the value specified by the *Number* variable. In the absence of the *Number* variable, the trap, if any, is removed. The *Macro* variable value must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **-**

.da [*Macro*]
Diverts, appending to the specified macro and appends version of the **.di** request. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level.

Initial Value: -

If No Value Specified: End current diversion

.de *Macro1* [*Macro2*]
Defines or redefines the value specified by the *Macro1* variable. The contents of the macro begins on the next input line. Input lines are copied in copy mode until the definition is stopped by a line beginning with *.Macro2*. In the absence of the *Macro2* variable, the definition is stopped by a line beginning with "**..**". A macro can contain **.de** requests, provided the stopping macros differ or the contained definition terminator is concealed. The "**..**" can be concealed as "**** .", which copies as "**\...**" and is reread as "**..**". The *Macro1* and *Macro2*variables must each be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **.***Macro2***=..**

.di [*Macro*]
Diverts output to the specified macro. Normal text processing occurs during diversion except that page offsetting is not performed. The diversion ends when the **.di** or **.da** request is encountered without a variable. Extraneous requests of this type should not be displayed when nested diversions are being used. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level.

Initial Value: **-**

If No Value Specified: End

.ds *StringName String*
Defines a string specified by the *StringName* variable to contain the value specified by the *String* variable. Any initial double-quote in *String* is stripped off to permit initial blanks. The *StringName* variable must be one or two ASCII characters.

.ds *StringName* ^A <*SetNumber*> <*MessageNumber*> [^A"<*DefaultMessage*> "] [^A<*Argument*> ^B<*Argument*> ^B <*Argument*>...]

Provides an alternate **.ds** syntax that allows the use of a message catalog for language-independent string definitions.

Based on the message *SetNumber* and the *MessageNumber* within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is placed into the *StringName* variable. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A (**^A**) delimits message identification, default message and optional argument list. The ASCII code Control-B (**^B**) delimits an individual optional argument list.

In the following example,

```
.ds {c ^A2 41^A"ERROR: (%1$s) input line \
%2$s" ^A\n(.F^B\n(.c
```

`2` is the message set number.

`41` is the message number.

text within quotes (`". . ."`) is the default message.

`\n(.F` is the name of the current input file.

`\n(.c` is the number of lines read from the input file.

If you assume the **troff** command runs with these conditions:

- The message at set 2 and number 41 matches the default message
- The current input file is `paper.doc`
- The **.ds** directive is on line 124 in the input file.

then the string `{c` would be defined as:

```
ERROR: (paper.doc)input line 123
```

Other examples are:

```
.ds {c ^A2 41
/* Without optional default message */
```

```
.ds {c ^A2 41^A"ERROR: (%1$s) input file \
%2$s" /* Without optional arguments */
```

If both the set number and the message number are set to zero, then the current date is returned in the current local's format. A user defined date format string can be defined in the default message field. The user defined format string must conform to the conversion specifications outlined by the **strftime** function in *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions*.

In the following examples:

```
.ds DT^A0 0
```

If the current date were July 10, 1991, in an English U.S. locale, `DT` would be defined as `7/10/91`.

```
.ds DT^A0 0^A"Today is %B %d, %Y"
```

If the current date were July 10, 1991, in an English U.S. locale, `DT` would be defined as `Today is July 10, 1991`.

The second syntax method is not intended for general use. It is used in the **nroff** and **troff** macro files supplied with the system to facilitate internationalization of internally generated messages.

Initial Value: **-**

If No Value Specified: Ignored

.dt *Number Macro*

Installs a diversion trap at the position specified by the *Number* variable in the current diversion to start the specified macro. Another **.dt** request redefines the diversion trap. If no variables are given, the diversion trap is removed. The *Macro* variable must be one or two ASCII characters. Mode or relevant values are associated with the current diversion level.

Initial Value: **-**

If No Value Specified: Off

.em *Macro*

Calls the specified macro when all input has ended. The effect is the same as if the contents of the specified macro had been at the end of the last file processed. The specified macro must be one or two ASCII characters.

Initial Value: None

If No Value Specified: None

.it *Number Macro*

Sets an input-line-count trap to call the specified macro after the number of lines of text input specified by the *Number* variable have been read (control or request lines are not counted). The text can be inline text or text provided by macros called explicitly (through inline calls) or implicitly (through traps). The *Macro* variable must be one or two ASCII characters. Relevant values are part of the current environment.

Initial Value: **-**

If No Value Specified: Off

.rm *Name*

Removes the specified request, macro, or string. The *Name* variable value is removed from the name list and any related storage space is freed. Subsequent references have no effect. The *Name* variable must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: Ignored

.rn *Name1 Name2*

Renames the request, macro, or string value specified by the *Name1*variable to the value specified by the *Name2* variable. The *Name1*and *Name2* variable values must each be one or two ASCII characters.

Initial Value: Ignored

If No Value Specified: **-**

.wh *Number Macro*

Installs a trap to call the specified macro at the page position specified by the *Number* variable. A negative *Number* variable value is interpreted with respect to the page bottom. Any macro previously planted at the page position specified by the *Number* variable is replaced by the *Macro* variable value. A *Number* variable value of 0 refers to the top of a page. In the absence of the *Macro* variable, the first trap found at the page position specified by the *Number* variable, if any, is removed. The *Macro* variable must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **-**

Number Registers

.af *Register Indicator*

Assigns the format as specified by the *Indicator* variable to the specified register. The *Register* variable must be one or two ASCII characters. The available format *Indicator* variable values are as follows:

Indicator	Numbering Sequence
1	0,1,2,3,4,5, . . .
001	000,001,002,003,004,005, . . .
i	0,i,ii,iii,iv,v, . . .
I	0,I,II,III,IV,V, . . .
a	0,a,b,c, . . . ,z,aa,ab, . . . ,zz,aaa, . . .
A	0,A,B,C, . . . ,Z,AA,AB, . . . ,ZZ,AAA, . . .

An Arabic format indicator having *N* digits (for example, 000000001) indicates a field width of *N* digits. The read-only registers and the width function are always Arabic.

Japanese Language Support**:** The following value specifies the character width for formatting Japanese numeric output in kanji:

k The number is formatted as a kanji string. If this is requested when a non-kanji codeset is specified, a warning message is printed and the **1** format is used.

Initial Value: Arabic

If No Value Specified: **-**

.nr *Register +/-Number1 Number2*

Assigns the specified register the value specified by the *+/-Number* variable with respect to the previous value, if any. The increment for auto-incrementing is set to the *Number2* variable value. The *Register* variable must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **-**

.rr *Register*

Removes the specified register. If many registers are being created dynamically, it can become necessary to remove registers that are not needed to recapture internal storage space for new registers. The *Register* variable must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **-**

Tabs, Leaders, and Fields

.fc *Delimiter Indicator*

Sets the field delimiter to the specified delimiter; the padding indicator is set to the space character or to the specified indicator. In the absence of variables, the field mechanism is turned off. The *Delimiter* variable value and the *Indicator* variable value must be ASCII characters.

Initial Value: Off

If No Value Specified: Off

.lc *Character*

Sets the leader repetition character to the specified character, or removes specifying motion. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment.

Initial Value: **.**

If No Value Specified: None

.ta *Stop* [*Type*]*...*

Sets tab stops. Default tab stops are set at every eight characters for the **nroff** command and every half inch for the **troff** command. Multiple *StopType* pairs can be specified by separating them with spaces; a value preceded by **+** (plus sign) is treated as an increment to the previous stop value.

The specified type determines how the text is adjusted at the tab stops. The *Type* variable values are as follows:

Type	**Adjustment**
R	Right-adjusting
C	Centering
blank	Left-adjusting

Relevant values are part of the current environment.

Initial Value: 8 ens for the **nroff** command and 0.5 inch for the **troff** command

If No Value Specified: None

.tc *Character*

Sets the tab repetition character to the specified character, or removes specifying motion. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment.

Initial Value: None

If No Value Specified: None

Input/Output Conventions and Character Translations

.cc *Character*

Sets the basic control character to the specified character, or resets to "**.**". The *Character* variable value must be an ASCII character. Relevant values are part of the current environment.

Initial Value: **.**

If No Value Specified: **.**

.cu [*Number*]

A variant of the **.ul** request that causes every character to be underlined and causes no line breaks to occur in the affected input lines. That is, each output space following a **.cu** request is similar to an unpaddable space. The **.cu** request is identical to the **.ul** request in the **troff** command. Relevant values are part of the current environment.

Initial Value: Off

If No Value Specified: *Number*=**1**

.c2 *Character*

Sets the no-break control character to the specified character or resets to " ' ". The *Character* variable value must be an ASCII character. Relevant values are part of the current environment.

Initial Value: '

If No Value Specified: '

.ec *Character*

Sets the escape character to **** (backslash) or to the value specified by the *Character* variable, if given. The *Character* variable value must be an ASCII character.

Initial Value: ****

If No Value Specified: ****

.eo

Turns off the escape mechanism.

Initial Value: On

If No Value Specified: **-**

.lg [*Number*]

Turns on the ligature mode if the *Number* variable value is absent or nonzero; turns off ligature mode if the *Number* variable value is 0. If the *Number* variable value is 2, only the two-character ligatures are automatically called. The ligature mode is inhibited for request, macro, string, register, or file names, and in the copy mode. This request has no effect in the **nroff** command.

Initial Value: On, for the **troff** command

If No Value Specified: On

.tr *Character1 Character2 Character3 Character4*

Translates, among other things, the character value specified by the *Character1* variable into the *Character2* variable value, the character value specified by the *Character3* variable into the *Character4*variable value. If an odd number of characters is given, the last one is mapped into the space character. To be consistent, a particular translation must stay in effect from input to output time. All specified characters must be ASCII characters. To reset the **.tr** request, follow the request with previous variables given in duplicate.

For example, the following **.tr** request:

```
.tr aAbBc<C,>
```

can be reset by entering:

```
.tr aabbcc
```

It must stay in effect until logical output.

Initial Value: None

If No Value Specified: **-**

.ul [*Number*]

Underlines in the **nroff** command (or italicizes in the **troff**command) the number of input-text lines specified by the *Number*variable. Actually switches to underline font, saving the current font for later restoration. Other font changes within the span of a **.ul** request take effect, but the restoration undoes the last change. Output generated by the **.tl** request is affected by the font change, but does not decrement the *Number* variable value. For more information, refer to the section ”Three-Part Titles“. If the specified number is greater than 1, there is the risk that a trap-called macro can provide text lines within the span; environment switching can prevent this.

Relevant values are part of the current environment.

Initial Value: Off

If No Value Specified: *Number*=**1**

.uf *Font*

Underlines the font set to the value specified by the *Font* variable. In the **nroff** command, the *Font* variable cannot be on position 1 (initially Times Roman). The *Font* variable value must be an ASCII font name.

Initial Value: Italic

If No Value Specified: Italic

Hyphenation

.hc *Character*
Sets the hyphenation indicator character to the value specified by the *Character* variable or to the default. The indicator is not displayed in the output. The *Character* variable value must be an ASCII character. Relevant values are part of the current environment.

Initial Value: **\%**

If No Value Specified: **\%**

.hw *Word1...*
Specifies hyphenation points in words with embedded minus signs. Versions of a word with a terminal s are implied; that is, *dig-it* implies *dig-its*. This list is examined initially and after each suffix stripping. The space available is 1024 characters, or about 50 to 100 words.

Initial Value:

If No Value Specified: Ignored

.hy *Number*
Turns on automatic hyphenation if the specified number is equal to or greater than 1; turns it off if the specified number is equal to 0 (equal to the **.nh** request). If the specified number is 2, the last lines (ones that cause a trap) are not hyphenated. If the specified number is 4 or 8, the last or first two characters, respectively, of a word are not split off. These values are additive; for example, a value of 14 calls all three restrictions (number equal to 2, number equal to 4, and number equal to 8).

Relevant values are part of the current environment.

Initial Value: No hyphenation

If No Value Specified: Hyphenate

.nh
Turns off automatic hyphenation. Relevant values are part of the current environment.

Initial Value: No hyphenation

If No Value Specified: **-**

Three-Part Titles

.lt [+/-][*Number*]
Sets the length of title value specified by the +/-*Number* variable. The line length and the title length are independent. Indents do not apply to titles, although page offsets do. Relevant values are part of the current environment.

Initial Value: 6.5 inches

If No Value Specified: Previous

.pc *Character*
Sets the page number character to the specified character or removes it. The page-number register remains %. The *Character* variable value must be an ASCII character.

Initial Value: **%**

If No Value Specified: Off

.tl *'Left'Center'Right'*
The strings represented by the *Left*, *Center*, and *Right* variables, respectively, are left-adjusted, centered, and right-adjusted in the current title length. Any of the strings can be empty, and overlapping is permitted. If the page-number character (initially **%**) is found within any of the fields, it is replaced by the current page number having the format assigned to the **%** register. Any ASCII character that is not displayed in the strings can be used as the string delimiter.

Initial Value: **-**

If No Value Specified: **-**

Output-Line Numbering

.nm [+/-] [*Number*] [*M*] [*S*] [*I*]
Turns on line-number mode. If the *M* variable is specified, only those line numbers that are multiples of the *M* variable value are to be printed. Every line number is printed if the *M* variable is absent (default is *M*=**1**). When line-number mode is in effect, a three-digit Arabic number plus a digit space are prepended to output text lines. The text lines are thus offset by four digit spaces, but otherwise retain their line length. If the *S* variable is given, it specifies the number of digit spaces to be displayed between the line number and the text (default is *S*=**1**). If the *I* variable is given, it specifies the number of digit spaces to indent before the line number (default is *I*=**0**).

Relevant values are part of the current environment.

Initial Value: **-**

If No Value Specified: Off

.nn *Number*
Suspends line numbering. The specified number of lines are not numbered. Relevant values are part of the current environment.

Initial Value: **-**

If No Value Specified: *Number*=**1**

Conditional Acceptance of Input

The *Condition* variable specifies one of the following one-character names:

o — If the current page number is odd.

e — If the current page number is even.

t — If the formatter is the **troff** command.

n — If the formatter is the **nroff** command.

.if *Condition Anything* — If the value specified by the *Condition* variable is true, accepts the value specified by the *Anything* variable as input; in multiline case, uses **\{***Anything***\}**.

.if *!Condition Anything* — If the value specified by the *Condition* variable is false, accepts the value specified by the *Anything* variable as input.

.if *Number Anything* — If the expression states that the *Number* variable value is greater than 0, accept the value specified by the *Anything* variable as input.

.if *!Number Anything* — If the expression states that the *Number* variable value is less than or equal to 0, accepts the value specified by the *Anything* variable as input.

.if *'String1'String2' Anything* — If the *String1* variable value is identical to the *String2* variable value, accepts the value specified by the *Anything* variable as input. Any nonblank ASCII character not in the *String1* and *String2* variables can be used as the delimiter.

.if *!'String1'String2' Anything* — If the *String1* variable value is not identical to the *String2*variable value, accepts the value specified by the *Anything* variable as input. Any nonblank ASCII character not in the *String1* and *String2* variables can be used as the delimiter.

.el *Anything* — Specifies the **else** portion of an **if/else** conditional.

.ie *Condition Anything* — Specifies the **if** portion of an **if/else** conditional dependent on the value of the *Condition* variable. Can be used with any of the preceding forms of the **.if** request.

Environment Switching

.ev *Environment*
Switches to the specified environment. The value specified by the *Environment* variable must be 0, 1, or 2. Switching is done in push-down fashion so that restoring a previous environment must be performed with the **.ev** request rather than with a specific reference.

Initial Value: *Environment*=0

If No Value Specified: Previous

Insertions from Standard Input

.ex Exits from the **nroff** command or **troff** command. Text processing is stopped exactly as if all input had ended.

Initial Value: **-**

If No Value Specified: **-**

.rd *Prompt* Reads insertion from standard input until two newline characters in a row are found. If the standard input is the user's keyboard, the specified prompt (or the ASCII BEL character) is written onto the user's terminal. The **.rd** request behaves like a macro, and additional variables can be placed after the *Prompt* variable.

Initial Value: **-**

If No Value Specified: *Prompt*=the ASCII BEL character

Input and Output File Switching

.cf *File* Copies the contents of the specified file, uninterrupted, into the **troff** command output file at this point. Problems occur unless the motions in the file restore the current horizontal and vertical position.

Initial Value: **-**

If No Value Specified: **-**

.lf *Number File* Corrects the **troff** command interpretation of the current line number (as specified by the *Number* variable) and the current file (as specified by the *File* variable) for use in error messages.

Initial Value: **-**

If No Value Specified: **-**

.nx *File* Uses the specified file as the input file. The current file is considered ended and the input is immediately switched to the specified file.

Initial Value: **-**

If No Value Specified: End of file

.pi *Program* Pipes output to the specified program. This request must occur before any printing occurs. No variables are transmitted to the specified program.

Initial Value: **-**

If No Value Specified: **-**

.so *File* Switches the source file. The top input (file-reading) level is switched to the specified file. When this file ends, input is again taken from the original file. The **.so** request can be nested.

When a **.so** request is encountered, the processing of the specified file is immediate. Processing of the original file (for example, a macro that is still active) is suspended.

A file should be preprocessed, if necessary, before being called by the **.so** request. The **eqn**, **tbl**, **pic**, and **grap** commands do not reach through a **.so** request to process an object file.

Initial Value: **-**

If No Value Specified: **-**

Miscellaneous

.ab *Text*

Prints the value specified by the *Text* variable to the diagnostic output (usually the terminal) and ends without further processing. If text is missing, the message `User Abort` is printed and the output buffer is flushed. This request is used in interactive debugging to force output.

.ab ^A<*SetNumber*> <*MessageNumber*> [^A"<*Default*> "] [^A<*Argument*> ^B<*Argument*> ^B<*Argument*>...]

Provides alternate syntax to allow use of a message catalog for language-independent abort messages. Prints the appropriate message specified by the parameter on the diagnostic output (usually the terminal) and ends without further processing. If there are no parameters, the message catalog equivalent to the following:

```
troff: User Abort, line no. file filename
```

is output. The output buffer is flushed. This request is used in interactive debugging to force output.

Based on the message *SetNumber* and the *MessageNumber* variables within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is written to the user's terminal. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A (`^A`) delimits message identification, default message, and optional argument list. The ASCII code Control-B (`^B`) delimits individual optional argument list.

In the following example:

```
.ab ^A2 42^A"Processing has been terminated \
 at line %1$s."^A\n(c.
```

`2` is the message set number.

`42` is the message number.

Text within quotes `"..."` is the default message.

`\n(c.` is the number of lines read from the input file.

If you assume the **troff** command runs with the following conditions:

- The message at set `2` and number `42`matches the default message.
- The **.ab** directive is on line `124`in the input file.

then the following would be displayed on the user's terminal:

```
Processing has been terminated at line 123.
```

Initial Value: **-**

If No Value Specified: User cancel

.Dt *Parameter*

Defines the format for returning the date within the **nroff** or **troff** request. By default, without the optional *Parameter*, the locale-specific date format specified by the current locale setting for the **LC_TIME** category is used. This corresponds to the "%x" format specifier of **strftime**. *Parameter* is a format string identical to the format string used with the **strftime** function in *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions*. Reference this function for a complete list of the format specifiers.

For example,

```
.Dt "%A, %B %d, %Y (%T)"
```

provides the following output for an English-speaking locale:

```
Thursday, January 31, 1991 (10:40:00)
```

The %A format is replaced by the locale-specific weekday name. The %B format is replaced by the locale-specific month name. The %d format is replaced by the day of the month in a two-digit format. The %Y format is replaced by the year with the century as a decimal number. The %T format is replaced by the time in hours (24-hour clock), minutes, and seconds in decimal numbers. This format provides for leap seconds and double leap seconds.

.fl

Flushes output buffer. This request usually causes a line break similar to the **.br** request. Calling this request with the control character " ' " (instead of "**.**") suppresses that break function.

Initial Value: **-**

If No Value Specified: **-**

.ig *Macro*

Ignores input lines. The **.ig** request works exactly like the **.de** request, except that the input is discarded. For more information, refer to "Macros, Strings, Diversions, and Position Traps". The input is read in copy mode, and any auto-incremented registers are affected. The *Macro* variable must be one or two ASCII characters.

Initial Value: **-**

If No Value Specified: **.***Macro*=**..**

.mc [*Character*] [*N*]

Uses the specified character as the margin character to display the specified distance (*N*) to the right of the margin after each non-empty text line (except those produced by the **.tl** request). If the output line is too long (as can happen in no-fill mode), the character is appended to the line. If the *N* variable is not given, the previous *N* variable is used. The first *N* variable is 0.2 inches in the **nroff** command and 1 em in the **troff** command.

Relevant values are part of the current environment.

Initial Value: .2 inches in **nroff**; 1 em in **troff**

If No Value Specified: Off

.pm [*Character*]

Prints macros. The names and sizes of all of the defined macros and strings are printed on the user's terminal. If any ASCII alphanumeric character is given as a variable, only the total of the sizes is printed. The size is given in blocks of 128 characters.

Initial Value: **-**

If No Value Specified: All

.sy *Command* [*Flags*]

The specified command is run but its output is not captured at this point. The standard input for the specified command is closed. Output must be explicitly saved in an output file for later processing. Often the **.sy** directive is followed by a subsequent **.so** directive to include the results of the previous command.

For example:

```
.sy date > /tmp/today
Today is
.so /tmp/today
```

Initial Value: **-**

If No Value Specified: **-**

.tm *String*

The specified string is written to the user's terminal.

.tm ^A<*SetNumber*> <*MessageNumber*> [^A"<*DefaultMessage*> "] [^A<*Argument*> ^B <*Argument*> ^B<*Argument*> ...]

Based on the message set number and the message number within the locale-specific catalog, the message catalog is read in copy mode and the corresponding message is written to the user's terminal. The initial sequence specifying the message set and message number can be omitted for backward compatibility. The ASCII code Control-A ^A delimits message identification, default message, and optional argument list. The ASCII code Control-B ^B delimits individual optional argument list.

In the following example:

```
.tm ^A2 23^A"The typesetter is %1$s.On line
%2$s."^A\*(.T^B\n(c.
```

2 is the message set number.

23 is the message number.

Text within quotes "..." is the default message.

*(.T is the first argument in troff for value of **-T**.

\n(c. is the number of lines read from the input file.

If you assume the **troff** command runs with the following conditions:

- The message at set 2 and number 23 matches the default message.
- The command line has **troff** using the **-T** option with device PSC.
- The **.tm** directive is on line 539 in the input file.

Then the following would be displayed on the user's terminal:

```
The typesetter is psc. On line 538.
```

The locale-specific message catalog is found in **/usr/lib/nls/msg/$LANG/macros.cat**.

Initial Value: **-**

If No Value Specified: Newline

Notes

The following notes apply to the **nroff** and **troff** requests. They are referenced by number in the requests where they apply.

1. The **.L** string register contains the current program locale value of all the categories.
2. The **.m** string register contains the locale value of the **LC_MESSAGES** category.
3. The **.t** string register contains the locale value for the **LC_TIME** category.
4. While the **.L**, **.t**, and **.m**string registers provide access to some environment values, a more general technique can be used to access any other environment variable. For example, if the **TED** environment variable is exported, the following **troff** commands:

```
.sy echo .ds z $TED >x
.so x
.sy rm x
```

set the *z* string register to contain the value of **$TED**.

Environment Variables

LC_ALL	Specifies the locale to be used for all the locale categories. It overrides any setting of the other locale environment variables.
LC_MESSAGES	Specifies the locale value for the **LC_MESSAGES** category. This is used if the **LC_ALL** environment variable is not set.
LC_TIME	Specifies the locale value for the **LC_TIME** category. This is used if the **LC_ALL** environment variable is not set.
LANG	Specifies the locale value to be used for all the locale categories. This is used if none of the above environment variables are set. This is the most often used environment variable to specify the locale.

Files

/usr/share/lib/tmac/tmac.*	Contains the pointers to standard macro files.
/usr/share/lib/macros/*	Denotes standard macro files.
/usr/share/lib/tmac/tmac.an	Contains the pointer to the **man** macro package.
/usr/share/lib/macros/an	Contains the **man** macro package.
/usr/share/lib/tmac/tmac.e file	Contains the **me** macro definition file.
/usr/share/lib/me directory	Contains the macro definition files.
/usr/share/lib/tmac/tmac.m	Contains the pointer to the **mm** macro package.
/usr/share/lib/macros/mmn	Contains the **mm** macro package.
/usr/share/lib/macros/mmt	Contains the **mm** macro package.
/usr/share/lib/tmac/tmac.ptx	Points to the macro package.
/usr/share/lib/macros/ptx	Contains the macro package.
/usr/share/lib/tmac/tmac.x	Contains the macro definition files.
/usr/share/lib/ms	Contains the **ms** macro definitions.
/usr/share/lib/tmac/tmac.v	Contains macro definitions.
/usr/share/lib/macros/vmca	Contains macro definitions.
/usr/lib/nls/msg/$LANG/macros.cat	Contains locale-specific message catalog for the **mm**, **me**, **ms**, and **mv** macro packages.
/usr/lib/font/dev*/*	Contains the font width tables.
/var/tmp/trtmp*	Denotes a temporary file.

Related Information

The **col** command, **eqn** command, **grap** command, **hplj** command, **ibm3812** command, **ibm3816** command, **mm** command, **mmt** command, **mvt** command, **neqn** command, **nroff** command, **pic** command, **ptx** command, **refer** command, **tbl** command, **tc** command, **xpreview** command.

The **nroff** and **troff** Input file format, **troff** file format, **troff** font file format.

The **setlocale** function, **strftime** function.

Message Facility in in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

National Language Support Overview in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

trpt Command

Purpose

Performs protocol tracing on TCP sockets.

Syntax

trpt [**-a**] [**-f**] [**-j**] [**-p***Address*]... [**-s**] [**-t**]

Description

The **trpt** command queries the buffer for Transmission Control Protocol (TCP) trace records. This buffer is created when a socket is marked for debugging with the **setsockopt** subroutine. The **trpt** command then prints a description of these trace records.

Note: You can use the **traceson** command to turn on socket level debugging for daemons.

When you specify no options, the **trpt** command prints all the trace records found in the system and groups them according to their TCP/IP connection protocol control block (PCB).

Before you can use the **trpt** command, you must:

1. Isolate the problem and mark for debugging the socket or sockets involved in the connection.
2. Find the address of the protocol control blocks associated with these sockets by using the **netstat -aA** command.
3. Then you can run the **trpt** command, using the **-p** flag to supply the associated protocol control block addresses. You can specify multiple **-p***Address* flags with a single **trpt** command.

The **-f** flag can be used to follow the trace log once it is located. The **-j** flag can be used to check the presence of trace records for the socket in question.

If the system image does not contain the proper symbols to find the trace buffer, the **trpt** command cannot succeed.

Output Fields

The information put out by the **trpt** command varies with the flag you use. Definitions of the fields contained in the various types of output follow:

Protocol Control Block identifier Identifies the protocol block to be traced, as shown in the following example:

```
4c500c:
```

Timestamp Specifies the time at which the connection is attempted, as shown in the following example:

```
500
```

Connection State

Specifies the state of the connection with the protocol control block:

CLOSED
Connection is closed.

LISTEN
Listening for a connection.

SYN_SENT
Active; have sent SYN. Represents waiting for a matching connection request after having sent a connection request.

SYN_RCVD
Have sent and received SYN. Represents waiting for a confirming connection request acknowledgment after having both received and sent connection requests.

ESTABLISHED
Connection established.

CLOSE_WAIT
Have received FIN; waiting to receive CLOSE.

LAST_ACK
Have received FIN and CLOSE; awaiting FIN ACK.

FIN_WAIT_1
Have closed; sent FIN.

CLOSING
Closed; exchanged FIN; awaiting FIN.

FIN_WAIT_2
Have closed; FIN is acknowledged; awaiting FIN.

TIME_WAIT
In 2MSL (twice the maximum segment length) quiet wait after close.

Action

Specifies the current status of the packet trace connection. The output of the command changes depending on the action.

Input Receiving input packets. The syntax of the output is:

```
input (SourceAddress, Port, DestinationAddress,
Port) <Sequence Number of the First Data Octet> @
AcknowledgementNumber
```

as in the following example:

```
input (src=129.353173176,23, dst=129.35.17.140, 1795) fb9f5461@fb9e4c68
```

Output Transmitting packets. The syntax of the output is:

```
output (SourceAddress, Port, DestinationAddress,
Port) <Sequence Number Of The First Data Octet>..
<Sequence Number of the Last Data Octet>@
AcknowledgementNumber)
```

as in the following example:

```
output (src=129.35.17.140,1795, dst=129.35.17.176, 23) fb9e4c68@fb9f5462
```

Window Size
Specifies the size of the window sending or receiving packets, as shown in the following example:

```
(win=1000)
```

User Specifies user request. The following is an example of a user request:
`SLOWTIMO<KEEP>`

Types of user requests and their definitions follow:

PRU_ATTACH Attach protocol to up.

PRU-DETACH Detach protocol from up.

PRU_BIND Bind socket to address.

PRU_LISTEN Listen for connection.

PRU_CONNECT
Establish connection to peer.

PRU_ACCEPT Accept connection from peer.

PRU_DISCONNECT
Disconnect from peer.

PRU_SHUTDOWN
Will not send any more data.

PRU_RCVD Have taken data; more room now.

PRU_SEND Send this data.

PRU_ABORT Abort (fast DISCONNECT, DETACH).

PRU_CONTROL
Control operations on protocol.

PRU_SENSE Return status into m.

PRU_RCVOOB Retrieve out of band data.

PRU_SENDOOB
Send out of band data.

PRU_SOCKADDR
Fetch socket's address.

PRU_PEERADDR
Fetch peer's address.

PRU_CONNECT2
Connect two sockets.

PRU_FASTTIMO
200 milliseconds timeout.

PRU_SLOTIMO 500 milliseconds timeout.

PRU_PROTORCV
Receive from below.

PRU_PROTOSEND
Send to below.

Drop Specifies that data was in preceding segment; data is dropped.

Window and Sequence Variables

Types of window and sequence variables follow:

rcv_nxt Next sequence number expected on incoming segments.

rcv_wnd
Size of receive window.

snd_una
Oldest unacknowledged sequence number.

snd_nxt
Next sequence number to be sent.

snd_max
Highest sequence number sent.

snd_sl1
Window update segment sequence number.

snd_wl1
Window update segment ack number.

snd_wnd
Send window.

Flags

-a	Prints the values of the source and destination addresses for each packet recorded, in addition to the normal output.
-f	Follows the trace as it occurs, waiting briefly for additional records each time the end of the log is reached.
-j	Lists just the protocol control block addresses for which trace records exist.
-p*Address*	Shows only trace records associated with the protocol control block specified in hexadecimal by the *Address* variable. You must repeat the **-p** flag with each *Address* variable specified.
-s	Prints a detailed description of the packet-sequencing information, in addition to the normal output.
-t	Prints the values for all timers at each point in the trace, in addition to the normal output.

Examples

1. To print trace information as well as the source and destination addresses for each packet recorded, enter:

   ```
   $ trpt -a
   ```

   This might display the following output:

   ```
   124b0c:
   900 ESTABLISHED:input (src=192.9.201.3,4257, dst=192.9.201.2,102
   5)2326e6e5@ad938c02(win=200)<ACK,FIN,PUSH> -> CLOSE_WAIT
   900 CLOSE_WAIT:output (src=192.9.201.2,1025, dst=192.9.201.3,425
   7)ad938c02@2326e6e6(win=4000)<ACK> -> CLOSE_WAIT
   900 LAST_ACK:output (src=192.9.201.2,1025, dst=192.9.201.3,4257)
   ad938c02@2326e6e6(win=4000)<ACK,FIN> -> LAST_ACK
   900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
   900 LAST_ACK:user DETACH -> LAST_ACK 12500c:
   800 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
   2)ad8eaa13@2326e6e5(win=4000)<ACK> -> ESTABLISHED
   800 ESTABLISHED:input (src=192.9.201.3,512, \
   dst=192.9.201.2,1024)
   [2326e6e5..2326e727)@ad8eaa13(win=1ef)<ACK,PUSH> -> ESTABLISHED
   800 ESTABLISHED:user RCVD -> ESTABLISHED
   900 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
   2)ad8eaa13@2326e727(win=4000)<ACK> -> ESTABLISHED
   900 ESTABLISHED:input (src=192.9.201.3,512, \
   dst=192.9.201.2,1024)
   ```

```
[2326e727..2326e82f)@ad8eaa13(win=1ef)<ACK,PUSH> -> ESTABLISHED
900 ESTABLISHED:user RCVD -> ESTABLISHED
900 ESTABLISHED:output (src=192.9.201.2,1024, dst=192.9.201.3,51
2)ad8eaa13@2326e82f(win=4000)<ACK> -> ESTABLISHED
900 ESTABLISHED:input (src=192.9.201.3,512, \
dst=192.9.201.2,1024)
2326e82f@ad8eaa13(win=1ef)<ACK,FIN,PUSH> -> CLOSE_WAIT
900 CLOSE_WAIT:output (src=192.9.201.2,1024, \
dst=192.9.201.3,512)
ad8eaa13@2326e830(win=4000)<ACK> -> CLOSE_WAIT
900 LAST_ACK:output (src=192.9.201.2,1024, dst=192.9.201.3,512)a
d8eaa13@2326e830(win=4000)<ACK,FIN> -> LAST_ACK
900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
900 LAST_ACK:user DETACH -> LAST_ACK
$ _
```

2. To list the protocol control blocks that have trace records, enter:

   ```
   trpt -j
   ```

   This might display the following output:

   ```
   124b0c, 12500c
   ```

3. To print the trace records associated with a single protocol control block, enter:

   ```
   trpt -p 12500c
   ```

   This might display the following output:

   ```
   800 ESTABLISHED:output ad8eaa13@2326e6e5(win=4000)<ACK> ->
   ESTABLISHED
   800 ESTABLISHED:input [2326e6e5..2326e727)@ad8eaa13(win=1ef)
   <ACK,PUSH> -> ESTABLISHED
   800 ESTABLISHED:user RCVD -> ESTABLISHED
   900 ESTABLISHED:output ad8eaa13@2326e727(win=4000)<ACK> -> ESTABLISHED
   900 ESTABLISHED:input [2326e727..2326e82f)@ad8eaa13(win=1ef) <ACK,PUSH> -> ESTABLISHED
   900 ESTABLISHED:user RCVD -> ESTABLISHED
   900 ESTABLISHED:output ad8eaa13@2326e82f(win=4000)<ACK> -> ESTABLISHED
   900 ESTABLISHED:input 2326e82f@ad8eaa13(win=1ef)<ACK,FIN,PUSH> -> CLOSE_WAIT
   900 CLOSE_WAIT:output ad8eaa13@2326e830(win=4000)<ACK> -> CLOSE_WAIT
   900 LAST_ACK:output ad8eaa13@2326e830(win=4000)<ACK,FIN> -> LAST_ACK
   900 CLOSE_WAIT:user DISCONNECT -> LAST_ACK
   900 LAST_ACK:user DETACH -> LAST_ACK
   $ _
   ```

Related Information

The **netstat** command, **tracesoff** command, **traceson** command.

The **setsockopt** subroutine.

Transmission Control Protocol/Internet Protocol, TCP/IP Protocols, TCP/IP routing in *Networks and communication management*.

true or false Command

Purpose

Returns an exit value of zero (true) or a nonzero exit value (false).

Syntax

true

false

Description

The **true** command returns a zero exit value. The **false** command returns a nonzero exit value. These commands are most often used as part of a shell script.

Examples

To construct a loop that displays the date and time once each minute, use the following code in a shell script:

```
while true
do
   date
   sleep 60
done
```

Related Information

Creating and running a shell script in *Operating system and device management* provides information on creating and executing shell procedures.

Commands overview in *Operating system and device management*.

---

truss Command

Purpose

Traces a process's system calls, dynamically loaded user level function calls, received signals, and incurred machine faults.

Syntax

truss [**-f**] [**-c**] [**-a**] [**-l**] [**-d**] [**-D**] [**-e**] [**-i**] [{ **-t** | **-x**} [**!**] *Syscall* [...]] [**-s** [**!**] *Signal* [...]] [{ **-m** }[**!**] *Fault* [...]] [{ **-r** | **-w**} [**!**] *FileDescriptor* [...]] [{ **-u** } [**!**]*LibraryName* [...]:: [**!**]*FunctionName* [...]] [**-o** *Outfile*] {*Command*\ **-p** *pid* [. . .]}

Description

The **truss** command executes a specified command, or attaches to listed process IDs, and produces a trace of the system calls, received signals, and machine faults a process incurs. Each line of the trace output reports either the *Fault* or *Signal* name, or the *Syscall* name with parameters and return values. The subroutines defined in system libraries are not necessarily the exact system calls made to the kernel. The **truss** command does not report these subroutines, but rather, the underlying system calls they make. When possible, system call parameters are displayed symbolically using definitions from relevant system header files. For path name pointer parameters, **truss** displays the string being pointed to. By default, undefined system calls are displayed with their name, all eight possible argments and the return value in hexadecimal format.

When the **-o** flag is used with **truss**, or if standard error is redirected to a non-terminal file, **truss** ignores the hangup, interrupt, and signals processes. This facilitates the tracing of interactive programs which catch **interrupt** and **quit** signals from the terminal.

If the trace output remains directed to the terminal, or if existing processes are traced (using the **-p** flag), then **truss** responds to **hangup**, **interrupt**, and **quit** signals by releasing all traced processes and exiting. This enables the user to terminate excessive trace output and to release previously existing processes. Released processes continue to function normally.

For those options which take a list argument, the name **all** can be used as a shorthand to specify all possible members of the list. If the list begins with a **!**, the meaning of the option is negated (for example,

exclude rather than trace). Multiple occurrences of the same option may be specified. For the same name in a list, subsequent options (those to the right) override previous ones (those to the left).

Every machine fault, with the exception of a page fault, results in posting a signal to the process which incurred the fault. A report of a received signal immediately follows each report of a machine fault, unless that signal is being blocked by the process.

To avoid collisions with other controlling processes, **truss** does not trace a process which it detects is being controlled by another process with the **/proc** interface.

The trace output for multiple processes is not produced in strict time order. For example, a read on a pipe may be reported before the corresponding write. However, for each process the output is strictly time-ordered. The trace output contains tab characters and standard tab stops are set at every eight positions.

The system may run out of per-user process slots when tracing children. This is because when tracing more than one process, **truss** runs as one controlling process for each process being traced, doubling the number of process slots being used for any given process. The usual system-imposed limit of 25 processes per user should be taken into account prior to running a trace on multiple processes

The operating system enforces certain security restrictions on the tracing of processes. You must have access privileges to the commands you are tracing. The **set-uid** and **set-gid** processes can only be traced by a privileged user. The **truss** command loses control of any process which performs an execution of a set-id or unreadable object file, unless it is run by a privileged user. These untraced processes continue normally and independently of truss from the point of the execution.

The lightweight processes (LWP) mentioned in truss output are really kernel threads. The option **-l** displays the LWP id (i.e. the thread id) on each line of the trace output.

User library functions in AIX libraries have both static and dynamic loaded function calls. The tracing with option **-u** is done for dynamically loaded function calls only.

User level function call tracing for dynamically loaded function calls is provided with **-u** option. This option will produce an entry/exit trace of the function calls.

Flags

-a Displays the parameter strings which are passed in each exec system call.

-c Counts traced system calls, faults, and signals rather than displaying trace results line by line. A summary report is produced after the traced command terminates or when **truss** is interrupted. If the **-f** flag is also used, the counts include all traced Syscalls, Faults, and Signals for child processes.

-d A timestamp will be included with each line of output. Time displayed is in seconds relative to the beginning of the trace. The first line of the trace output will show the base time from which the individual time stamps are measured. By default timestamps are not displayed.

-D Delta time is displayed on each line of output. The delta time represents the elapsed time for the LWP that incurred the event since the last reported event incurred by that thread. By default delta times are not displayed.

-e Displays the environment strings which are passed in each exec system call.

-f Follows all children created by the **fork** system call and includes their signals, faults, and system calls in the trace output. Normally, only the first-level command or process is traced. When the **-f** flag is specified, the process id is included with each line of trace output to show which process executed the system call or received the signal.

-i Keeps interruptible sleeping system calls from being displayed. Certain system calls on terminal devices or pipes, such as **open** and **kread**, can sleep for indefinite periods and are interruptible. Normally, **truss** reports such sleeping system calls if they remain asleep for more than one second. The system call is then reported a second time when it completes. The **-i** flag causes such system calls to be reported only once, upon completion.

-l Display the id (thread id) of the responsible LWP process along with truss output. By default LWP id is not displayed in the output.

-m [!]*Fault* Traces the machine faults in the process. Machine faults to trace must be separated from each other by a comma. Faults may be specified by name or number (see the **sys/procfs.h** header file). If the list begins with the ″!″ symbol, the specified faults are excluded from being traced and are not displayed with the trace output. The default is **-m**`all` **-m**`!fltpage`.

-o *Outfile* Designates the file to be used for the trace output. By default, the output goes to standard error.

-p Interprets the parameters to **truss** as a list of process ids for an existing process rather than as a command to be executed. **truss** takes control of each process and begins tracing it, provided that the user id and group id of the process match those of the user or that the user is a privileged user.

-r [**!**] *FileDescriptor* Displays the full contents of the I/O buffer for each read on any of the specified file descriptors. The output is formatted 32 bytes per line and shows each byte either as an ASCII character (preceded by one blank) or as a two-character C language escape sequence for control characters, such as horizontal tab (\t) and newline (\n). If ASCII interpretation is not possible, the byte is shown in two-character hexadecimal representation. The first 12 bytes of the I/O buffer for each traced read are shown, even in the absence of the **-r** flag. The default is **-r!all**.

-s [**!**] *Signal* Permits listing *Signals* to trace or exclude. Those signals specified in a list (separated by a comma) are traced. The trace output reports the receipt of each specified signal even if the signal is being ignored, but not blocked, by the process. Blocked signals are not received until the process releases them. Signals may be specified by name or number (see sys/signal.h). If the list begins with the ″**!**″ symbol, the listed signals are excluded from being displayed with the trace output. The default is **-s all**.

-t [**!**] *Syscall* Includes or excludes system calls from the trace process. System calls to be traced must be specified in a list and separated by commas. If the list begins with an ″**!**″ symbol, the specified system calls are excluded from the trace output. The default is **-tall**.

-u [!] [*LibraryName* [...]::[!]*FunctionName* [...]]
: Traces dynamically loaded user level function calls from user libraries. The *LibraryName* is a comma-separated list of library names. The *FunctionName* is a comma-separated list of function names. In both cases the names can include name-matching metacharacters *****, **?**, **[]** with the same meanings as interpreted by the shell but as applied to the library/function name spaces, and not to files.

  A leading **!** on either list specifies an exclusion list of names of libraries or functions not to be traced. Excluding a library excludes all functions in that library. Any function list following a library exclusion list is ignored. Multiple **-u** options may be specified and they are honored left-to-right. By default no library/function calls are traced.

-w [**!**] *FileDescriptor*
: Displays the contents of the I/O buffer for each write on any of the listed file descriptors (see **-r**). The default is **-w!all**.

-x [**!**] *Syscall*
: Displays data from the specified parameters of traced sytem calls in raw format, usually hexadecimal, rather than symbolically. The default is **-x!all**.

Examples

1. To produce a trace of the **find** command on the terminal, type:
   ```
   truss find . -print >find.out
   ```
2. To trace the lseek, **close**, statx, and open system calls, type:
   ```
   truss -t lseek,close,statx,open find . -print > find.out
   ```
3. To display thread id along with regular output for **find** command, enter:
   ```
   truss -l find . -print >find.out
   ```
4. To display timestamps along with regular output for **find** command, enter:
   ```
   truss -d find . -print >find.out
   ```
5. To display delta times along with regular output for **find** command, enter:
   ```
   truss -D find . -print >find.out
   ```
6. To trace the **malloc()** function call and exclude the **strlen()** function call in the **libc.a** library while running the **ls** command, enter:
   ```
   truss -u libc.a::malloc,!strlen ls
   ```
7. To trace all function calls in the **libc.a** library with names starting with ″m″ while running the **ls** command, enter:
   ```
   truss -u libc.a::m*,!strlen ls
   ```
8. To trace all function calls from the library **libcurses.a** and exclude calls from **libc.a** while running executable `foo`, enter:
   ```
   truss -u libcurses.a,!libc.a::* foo
   ```
9. To trace the **refresh()** function call from **libcurses.a** and the **malloc()** function call from **libc.a** while running the executable `foo`, enter:
   ```
   truss -u libc.a::malloc -u libcurses.a::refresh foo
   ```

Files

/proc filesystem

Related Information

The **fork** subroutine.

The **/proc** file.

---

tset Command

Purpose

Initializes terminals.

Syntax

tset [**-e** *C*] [**-k** *C*] [**-i** *C*] [**-**] [**-s**] [**-l**] [**-Q**] [**-m** [*Identifier*] [*TestBaudRate*] **:***Type*] ... [*Type*]

Description

The **tset** command enables you to set the characteristics of your terminal. It performs terminal-dependent processing, such as setting erase and kill characters, setting or resetting delays, and sending any sequences needed to properly initialize the terminal.

The **tset** command first determines the type of terminal involved (specified by the *Type* parameter). It then performs necessary initializations and mode settings. The type of terminal attached to each port is specified in the Object Data Manager (ODM) database. The terminfo database contains possible type names for terminals. If a port is not wired permanently to a specific terminal (that is, it is not hardwired), the **tset** command gives it an appropriate generic identifier, such as `dialup`.

When no flags are specified, the **tset** command reads the terminal type out of the **TERM** environment variable and re-initializes the terminal.

When the **tset** command is used in a startup script (the **.profile** file for **sh** users or the **.login** file for **csh** users), the script should include information about the type of terminal you will usually use on ports that are not hardwired. These ports are identified in the ODM database as `dialup`, `plugboard`, or `ARPANET`, among others. To specify which terminal type you usually use on these ports, use the **-m** flag (followed by the appropriate port type identifier), an optional baud rate specification, and the terminal type. If more than one mapping is specified, the first applicable mapping prevails. A missing port type identifier matches all identifiers. Any of the alternate generic names given in the **terminfo** database can be used as the identifier.

You can specify the baud rate in the **tset** command as you would with the **stty** command. The baud rate is compared with the speed of the diagnostic output (which should be the control terminal). The baud rate test can be any combination of the following characters:

- `.` (period)
- `@` (at sign)
- < (less than sign)
- `!` (exclamation point)

The `@` (at sign) stands for the preposition at, and the `!` (exclamation point) inverts the sense of the test. To avoid problems with metacharacters, place the **-m** flag argument inside '' (single quotes). Users of the **csh** command must also put a \ (backslash) before any `!` (exclamation point).

The following example sets the terminal type to `adm3a` if the port in use is a dialup at a speed greater than `300` baud. It sets the terminal type to `dw2` if the port is a dialup port at a speed of `300` baud or less:

```
tset -m 'dialup>300:adm3a' -m dialup:dw2 -m 'plugboard:?adm3a'
```

If the *Type* parameter begins with a ? (question mark), you are prompted to verify the type. To use the specified type, press Enter. To use a different type, enter the type you want. In the example given, you are prompted to verify the `adm3` plugboard port type.

If no mapping applies and a final type option (not preceded by an **-m** flag) is given on the command line, that type is used. Otherwise, the default terminal type is the one identified in the ODM database. Hardwired ports should always be identified in the ODM database.

When the terminal type is known, the **tset** command engages in terminal driver mode setting. This usually involves setting:

- An initialization sequence to the terminal
- The single character erase and optionally the line-kill (full-line erase) characters
- Special character delays

Tab and new-line expansion are turned off during transmission of the terminal initialization sequence.

On terminals that can backspace but not overstrike (such as a CRT), and when the erase character is the default erase character (# on standard systems), the erase character is changed to Backspace (Ctrl-H).

Flags

-e *C*	Sets the erase character to the character specified by the *C* parameter.The default is the backspace character.
-I	Suppresses transmission of terminal initialization strings.
-i *C*	Sets the interrupt character to the character specified by the *C* parameter. The *C* parameter defaults to ^C (caret C). The ^ (caret) character can also be used for this option.
-k *C*	Sets the line-kill character to the character specified by the *C* parameter. The *C* parameter defaults to ^X (caret X). The ^ (caret) character can also be used for this option.
-m *IdentifierTestbaudRate*:*Type*	Specifies which terminal type (in the *Type* parameter) is usually used on the port identified in the *Identifier* parameter. A missing identifier matches all identifiers. You can optionally specify the baud rate in the *TestBaudRate* parameter.
-Q	Suppresses printing of the `Erase set to` and `Kill set to` messages.
-s	Prints the sequence of **csh** commands that initialize the **TERM** environment variable, based on the name of the terminal decided upon.
-	The name of the terminal decided upon is output to standard output. This is the **TERM** environment variable.

Examples

The following examples all assume the Bourne shell and usage of the **-** flag. If you use the **csh** command, use the preceding variations. A typical use of the **tset** command in a **.profile** or **.login** file includes the **-e** and **-k** flags, and often the **-n** or **-Q** flags as well. To streamline the examples, these flags have not been included here.

Note: Make sure to enter the **tset** command all on one line regardless of the number of lines used in the example.

1. Now you are a 2621 terminal. Do not use the following example in your **.profile** file, unless you are always a 2621 terminal.

   ```
   export TERM; TERM=\'tset \- 2621\'
   ```

2. You have an h19 terminal at home that you dial up on, but your office terminal is hardwired and specified in the ODM database.

```
export TERM; TERM=\'tset \- \-m dialup:h19"'
```

3. You have a switch that connects everything to everything, making it nearly impossible to key on what port you are coming in. You use a vt100 in your office at 9600 baud and dial up from home on a 2621 to switch ports at 1200 baud. Sometimes, you use a different terminal at work. At high speeds, you want to verify your terminal type, but at 1200 baud, you are always on a 2621. Note how the quotation marks protect the greater-than sign and the question mark from interpretation by the shell.

   ```
   export TERM; TERM=\'tset \- \-m 'switch>1200:?vt100' \-m
   'switch<=1200:2621'
   ```

   If none of the conditions hold, the terminal type specified in the ODM database is used.

4. The following entry is appropriate if you always dial up at the same baud rate on many different terminals. Your most common terminal is an `adm3a`. You are always prompted to verify the terminal type, which defaults to `adm3a`.

   ```
   export TERM; TERM=\'tset \- \?adm3a\'
   ```

5. If the ODM database is not properly installed and you want to key entirely on the baud rate, type:

   ```
   export TERM; TERM=\'tset \- \-m 'switch>1200:?vt100' \-m
   'switch<=1200:2621'
   ```

6. You dial up at 1200 baud or less on a Concept100, sometimes over switch ports and sometimes over regular dialups. You use various terminals at speeds higher than 1200 over switch ports, most often the terminal in your office, which is a vt100. However, sometimes you log in from the university over the ARPANET; in this case, you are on an ALTO emulating a dm2500. You also often log in on various hardwired ports, such as the console, all of which are properly entered in the ODM database. To set your erase character to Ctrl-H and your kill character to Ctrl-U, type:

   ```
   export TERM
   TERM=\'tset \-e \-k(hat)U \-Q \- "-m 'switch<1200:concept100'
   "-m 'switch:?vt100' \-m dialup:concept100 "1-m arpanet: dm2500"'
   ```

   This also prevents the **tset** command from printing the following line:

   ```
   Erase set to Backspace, Kill set to Ctrl-U
   ```

7. To set the erase character to a control character, type:

   ```
   tset -e ^Y
   ```

Files

/usr/share/lib/terminfo	Contains the terminal capability database.

Related Information

The **csh** command, **reset** command, **sh** command, **stty** command.

The **environ** file, **terminfo** file.

TERM Values for Different Displays and Terminals section of TTY terminal device in *Operating system and device management*.

---

tsh Command

Purpose

Invokes the trusted shell.

Syntax

Press in sequence: the Ctrl+X, Ctrl+R keys.

tsh Command

Description

The **tsh** command is a command interpreter that provides greater security than the Korn shell (the standard login shell). Generally, a user calls the **tsh** shell by pressing Ctrl+X, Ctrl+R, the secure attention key (SAK) sequence, after a login. The **tsh** shell also can be invoked by defining it as the login shell in the **/etc/passwd** file.

To use the SAK sequence to invoke the trusted shell, the terminal the user is using must have SAK enabled, and the user must be allowed to use the trusted path. See the Trusted Computing Base in *Operating system and device management* for information on enabling SAK on a terminal, and see the **/etc/security/user** file and the **chuser** command for information on allowing a user to access the trusted path.

To exit from the **tsh** shell, use any of the following commands: the **logout** command, **shell** command, **su** command. The **logout** command ends the login session, while the other commands execute the user's initial program and continue the login session.

The trusted shell differs from the Korn shell in the following ways:

- The function and alias definitions are not supported. Alias definitions are only supported in the **/etc/tsh_profile** file.
- The **IFS** and **PATH** environment variables cannot be redefined.
- Only trusted programs can be run from the **tsh** shell.
- The history mechanism is not supported.
- The only profile used is the **/etc/tsh_profile** file.
- The trusted shell has the following built-in commands:

logout Exits the login session and terminates all processes.
shell Re-initializes the user's login session. The effect is the same as logging in to the system.
su Resets the effective ID to the user's identity on the system and executes another trusted shell.

Security

Access Control: This command should be a standard user program and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	/etc/tsh_profile

Examples

To invoke the trusted shell, press the Ctrl+X, Ctrl+R key sequence, the secure attention key (SAK).

Files

/usr/bin/tsh Contains the **tsh** command.
/etc/tsh_profile Contains initialization commands for the trusted shell.

/etc/passwd	Contains basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/security/login.cfg	Contains configuration information.

Related Information

The **chuser** command, **init** command, **ksh** command, **logout** command, **shell** command, **su** command, **tsm** command.

See National Language Support Overview in *AIX 5L Version 5.3 National Language Support Guide and Reference* for more information about Single-Source Dual Object (SSDO) commands used during installation.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the Securing the network in *Security*.

See Trusted Computing Base in *Security* for more information about the trusted path and enabling SAK on a terminal.

---

tsm Command

Purpose

Provides terminal state management.

Syntax

tsm *Port*

Description

The **tsm** command invokes the terminal state manager, which controls the ports used in the trusted path. The functions are:

- Establishing line communication modes and discipline - functions performed by the **getty** command.
- Verifying the user's account and identity, and setting the initial process credentials and environment - functions performed by the **login** command.
- Performing trusted path management if the secure attention key (SAK) is enabled for the port and the system login program is used.

  Note: The **tsm** command is not entered on the command line.

Trusted path management occurs in two phases:

login This phase is in effect if a user has not successfully logged in. If the secure attention key (SAK) signal is detected, the system restarts getty-login type processing. The next login puts the user into the trusted state, if the port and the user support the trusted state.

shell This phase occurs after successful user authentication. The command functions according to the user's **tpath** attribute. The following values are valid:

on Provides standard trusted path management. When the secure attention key (SAK) signal is detected, all processes that access the port, except the **tsm** process and its siblings (including the trusted shell), are terminated the next time an attempt is made to access the port. The port is reset to its initial state and is marked as trusted, and the trusted shell command (the **tsh** command) is executed.

notsh The user session terminates when the secure attention key (SAK) signal is detected.

always The user is not allowed off the trusted path. The user's shell will always be the trusted shell, tsh.

nosak The secure attention key (SAK) is disabled for the terminal, and the user's initial program runs.

Security

Access Control: This command should grant execute (x) permission to any user. The command should be setuid to the root user and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/objrepos/CuAt**
r	**/usr/lib/objrepos/PdAt**
r	**/etc/security/login.cfg**
r	**/etc/security/user**

Examples

To provide terminal state management on `tty0`, add the following line to the **/etc/inittab** file:

```
tty0:2:respawn:/usr/sbin/tsm /dev/tty0
```

This initializes the port `/dev/tty0` and sets up the characteristics of the port.

Files

/usr/sbin/tsm	Contains the **tsm** command.
/etc/security/login.cfg	Contains configuration information.
/etc/security/user	Contains extended user attributes.

Related Information

The **getty** command, **init** command, **login** command, **logout** command, **setgroups** command, **shell** command, **su** command, **tsh** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the Securing the network in *Security*.

tsort Command

Purpose

Sorts an unordered list of ordered pairs (a topological sort).

Syntax

tsort [**—**] [*File*]

Description

The **tsort** command reads from *File* or standard input an unordered list of ordered pairs, builds a completely ordered list, and writes it to standard output.

The input *File* should contain pairs of non-empty strings separated by blanks. Pairs of different items indicate a relative order. Pairs of identical items indicate presence, but no relative order. You can use the **tsort** command to sort the output of the **lorder** command.

If *File* contains an odd number of fields, an appropriate error message is displayed.

Flag

—	(Double hyphen) Interprets all arguments following the — flag as file names. If the file is named —, use tsort — —.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Files

/usr/ccs/bin/tsort	Contains the **tsort** command.
/usr/ccs/bin/tsort	Contains symbolic link to the **tsort** command.

Related Information

The **ar** command, **ld** command, **lorder** command, **xargs** command.

The Commands overview in *Operating system and device management*.

---

ttt Command

Purpose

Starts the tic-tac-toe game.

Syntax

ttt [**-e**] [**i**]

Description

The **ttt** command starts the tic-tac-toe game. This is a learning version but it learns slowly. It loses nearly 80 games before completely mastering the game. When you start the game you are prompted `Accumulated knowledge? (Yes or No)`. Entering `y` provides the computer with knowledge gained from previous games.

You are always X and your opponent is always O. You can either make the first move or pass to your opponent. To pass, press the enter key when prompted `Your move?` at the beginning of the game. The first to get three in a row wins the game. For example:

```
new game
123
456
789
Your move?
1
XO3
456
789
Your move?
9
XOO
456
78X
Your move?
5
You win
```

In the example, your first move was to place an X where 1 was located. The computer placed an O where the 2 was located. The game progressed until you had three in a diagonal row (1,5, 9). The game repeats until you quit. To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence**.**

Flags

-e Increases the speed of the learning.
-i Displays the instructions prior to the start of the game.

Files

$HOME/ttt.a	Specifies the location of the learning file.
/usr/games	Specifies the location of the system's games.

Related Information

The **arithmetic** command, **back** command, **bj** command, **craps** command, **fish** command, **fortune** command, **hangman** command, **moo** command, **number** command, **quiz** command, **turnoff** command, **turnon** command, **wump** command.

---

tty Command

Purpose

Writes to standard output the full path name of your terminal.

Syntax

/usr/bin/tty [**-s**]

Description

The **tty** command writes the name of your terminal to standard output.

If your standard input is not a terminal and you do not specify the **-s** flag, you get the message `Standard input is not a tty`.

The following environment variables affect the execution of the **tty** command:

LANG	Determines the locale to use for the locale categories when neither the **LC_ALL** variable nor the corresponding environment variable beginning with **LC_** specifies a locale.
LC_ALL	Determines the locale to be used. This variable overrides any values for locale categories that are specified by any other environment variable beginning with **LC_** or by the **LANG** variable.
LC_CTYPE	Determines the locale for the interpretation of sequences of bytes of text data as characters. For example, this variable may specify multi-byte characters instead of single-byte characters.
LC_MESSAGES	Determines the language for messages.

Flags

-s Suppresses reporting the path name.

Exit Status

This command returns the following exit values:

0 Standard input is a terminal.
1 Standard input is not a terminal.
>1 An error occurred.

Examples

1. To display the full path name of your display:

   ```
   tty
   ```

2. To test whether or not the standard input is a terminal:

   ```
   if tty -s
   then
   echo 'Enter the text to print:' >/dev/tty
   qprt -
   fi
   ```

If the standard input is a terminal, this displays the message `"Enter the text to print:"` as a prompt and prints the text that the user types. If the standard input is not a terminal, this displays nothing; it merely prints the text read from the standard input.

The `echo . . . >/dev/tty` displays the prompt on the screen even if you redirect the standard output of the shell procedure. This way the prompt is never written into an output file. The special file **/dev/tty** always refers to your terminal, although it also has another name such as **/dev/console** or **/dev/tty2**.

Files

/usr/bin/tty	Contains the **tty** command.
/dev/tty	Specifies the **tty** pseudo device.

Related Information

The National Language Support Overview in *AIX 5L Version 5.3 National Language Support Guide and Reference* discusses the **LC_** variables.

tunchange Command

Purpose

Updates one or more tunable stanzas in a file.

Syntax

tunchange -f *Filename* (**-t** *Stanza* ({**-o** Parameter[=*Value*]} | **-D**) | **-m** *Filename2*)

Description

The **tunchange** command unconditionally updates a tunable file. It can also merge a second file with the current file.

Note: No message will be displayed (even when a parameter of type bosboot is changed).

Flags

-f *Filename*	Name of the updated tunable file. If the name does not include the '**/**' (forward slash) character, it is considered to be relative to **/etc/tunables**.
-t *Stanza*	Name of the stanza to update. *Stanza* is either **schedo**, **vmo**, **ioo**, **no**, **nfso**, or **raso**. *Stanza* corresponds to the name of the command which can update the parameter or parameters specified by the **-o** flag.
-o Parameter=*Value*	Parameter to be set to *Value*. It must be valid in the *Stanza* specified by the **-t** flag and consistent with the other parameters of the file specified by the **-f** flag.
-D	Resets all parameters of the *Stanza* to their default value.
-m *Filename2*	Merges the *Filename2* file with the current *Filename* file.

Exit Status

0	Changes were correctly applied.
>0	One of the following conditions caused an error: • The specified *Filename*, *Filename2*, or *Stanza* was invalid. • Parameter=*Value* was invalid for the Parameter. • No message was provided.

Examples

1. To update the **pacefork** parameter in the **/etc/tunables/nextboot** file, type:
   ```
   tunchange -f nextboot -t schedo -o pacefork=10
   ```
2. To update the **pacefork** parameter in the **/home/mine/mytunable** file, type:
   ```
   tunchange -f /home/mine/mytunable -t schedo -o pacefork=10
   ```
3. To reset all **schedo** stanza parameters to their default value in the **/etc/tunables/nextboot** file, type:
   ```
   tunchange -f nextboot -t schedo -D
   ```
4. To merge the **/home/mine/mytunable** file with the **/etc/tunables/nextboot** file, type:
   ```
   tunchange -f nextboot -m /home/mine/mytunable
   ```

Files

/usr/sbin/tunchange	Contains the **tunchange** command.
/etc/tunables/	Contains the default tunable files.

Related Information

The **raso**, **schedo**, **vmo**, **ioo**, **no**, **nfso**, **tuncheck**, **tundefault**, **tunrestore**, and **tunsave** commands.

Tunables File Format in *AIX 5L Version 5.3 Files Reference*.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*.

---

tuncheck Command

Purpose

Validates a tunable file.

Syntax

tuncheck [**-r** | **-p**] **-f** *Filename*

Description

The **tuncheck** command validates a tunable file. All tunables listed in the specified file are checked for range and dependencies. If a problem is detected, a warning is issued.

There are two types of validation:

against the current context
: Checks to see if *Filename* could be applied immediately. Tunables not listed in *Filename* are interpreted as current values. The checking fails if a tunable of type **Incremental** is listed with a smaller value than its current value; it also fails if a tunable of type **Bosboot** or **Reboot** is listed with a different value than its current value.

against the next boot context
: Checks to see if *Filename* could be applied during a reboot, that is, if it could be a valid nextboot file. Decreasing a tunable of type **Incremental** is allowed. If a tunable of type **Bosboot** or **Reboot** is listed with a different value than its current value, a warning is issued but the checking does not fail.

Additionally, warnings are issued if ***Filename*** contains unknown stanzas, or unknown tunables in a known stanza. However, that does not make the checking fail.

Upon success, the **AIX_level, Kernel_type** and **Last_validation** fields in the info stanza of the checked file are updated.

Flags

-f *Filename*	Specifies the name of the tunable file to be checked. If it does not contain the '**/**' (forward slash) character, the name is relative to **/etc/tunables**.
-p	Checks *Filename* in both current and boot contexts. This is equivalent to running **tuncheck** twice, one time without any flag and one time with the **-r** flag.
-r	Checks *Filename* in a boot context.

If **-p** or **-r** are not specified, *Filename* is checked according to the current context.

Tuning Parameter Types

Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Exit Status

0 *Filename* is valid.

>0 *Filename* is invalid, message have been provided.

Examples

1. To check whether **mytunable** can be applied immediately, type:
   ```
   tuncheck -f ./mytunable
   ```
2. To check whether **/etc/tunables/nextboot** can be applied during a reboot, type:
   ```
   tuncheck -r -f nextboot
   ```
3. To check whether **/etc/tunables/nextboot** can be applied immediately and after a reboot, type:
   ```
   tuncheck -p -f nextboot
   ```

Files

/usr/sbin/tunckeck	Contains the **tuncheck** command.
/etc/tunables	Contains all the tunable files.

Related Information

The **raso**, **schedo**, **vmo**, **ioo**, **no**, **nfso**, **tunchange**, **tundefault**, **tunrestore**, and **tunsave** commands.

Tunables File Format in *AIX 5L Version 5.3 Files Reference*.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*.

---

tundefault Command

Purpose

Reset all tunable parameters to their default value.

Syntax

tundefault [**-r** | **-p**]

Description

The **tundefault** command launches all the tuning commands (**ioo**, **vmo**, **schedo**, **no**, **nfso**, and **raso**) with the **-D** flag. This resets all the AIX tunable parameters to their default value, except for parameters of type **Bosboot** and **Reboot**, and parameters of type **Incremental** set at values bigger than their default value, unless **-r** was specified. Error messages are displayed for any parameter change impossible to make.

Flags

-p	Makes the changes permanent: resets all the tunable parameters to their default values and updates the **/etc/tunables/nextboot** file.
-r	Defers the reset to their default value to next reboot. This clears stanza(s) in the **/etc/tunables/nextboot** file, and if necessaty, proposes **bosboot** and warns that a reboot is needed

Tunable Parameter Types

Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Examples

1. To permanently reset all tunable parameters to their default values, type:

   ```
   tundefault -p
   ```

   All of the tuning commands are launched with the **-Dp** flags. This resets all the tunable parameters to their default value. This also updates the **/etc/tunables/nextboot** file. This command completely and permanently resets all tunable parameters to their default values.
2. To defer the setting of all tunable parameters until next reboot, type:

   ```
   tundefault -r
   ```

   Calls all tuning commands with **-Dr**. This clears all of the stanzas in the **/etc/tunables/nextboot** file, and if necessary, proposes **bosboot** and displays a message warning that a reboot is necessay to make the changes effective.

Files

/usr/sbin/tundefault	Contains the **tundefault** command.
/etc/tunables/	Contains all the tunable files.

Related Information

The **raso**, **schedo**, **vmo**, **ioo**, **no**, and **nfso** commands.

The “tunchange Command” on page 526, “tuncheck Command” on page 527, “tunrestore Command,” and “tunsave Command” on page 531.

Tunables File Format in *AIX 5L Version 5.3 Files Reference*.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*.

tunrestore Command

Purpose

Restores tunable parameter values from a file.

Syntax

turestore [**-r**] **-f** *Filename*

tunrestore -R

Note: tunrestore -R can only be called from **inittab**.

Description

The **tunrestore** command restores all tunable parameters values stored in a file.

tunrestore -f *Filename* immediatly applies *Filename*. All tunables listed in *Filename* are set to the value defined in this file. Tunables not listed in *Filename* are kept unchanged. Tunables explicitly set to `DEFAULT` are set to their default value.

tunrestore -r -f *Filename* applies *Filename* for the next boot. This is achieved by checking the specified file for inconsistencies (the equivalent of running **tuncheck** on it) and copying it over to **/etc/tunables/nextboot**. If bosboot is necessary, the user will be offered to run it.

tunrestore -R is only used during reboot. All tunables that are not already set to the value defined in the nextboot file are modified. Tunables not listed in the nextboot file are forced to their default value. All actions, warnings and errors are logged into **/etc/tunables/lastboot.log**.

Additionally, a new tunable file called **/etc/tunables/lastboot** is automatically generated. That file has all the tunables listed with numerical values. The values representing default values are marked with the comment DEFAULT VALUE. Its info stanza includes the checksum of the **/etc/tunables/lastboot.log** file to make sure pairs of lastboot/lastboot.log files can be identified.

Flags

-f *Filename*	Specifies the name of the tunable file to apply. If it does not contain the '**/**' (forward slash) character, the name is relative to **/etc/tunables**.
-r	Makes the specified file become the new **nextboot** file.
-R	Restores **/etc/tunables/nextboot** during boot process.

Tunable Parameter Types

Dynamic	Can be changed at any time.
Static	Can never be changed
Reboot	Can only be changed during the reboot sequence
Bosboot	Can only be changed by running bosboot and rebooting the machine
Mount	Changes made are only effective for future filesystems or directory mountings
Incremental	Can only be incremented, except at boot time.
Connect	Changes are only effective for future socket connections.

Examples

1. To restore all tunable values stored in **/etc/tunables/mytunable**, type:

   ```
   tunrestore -f mytunable
   ```

2. To validate **/etc/tunables/mytunable** and make it the new nextboot file, type:

   ```
   tunrestore -r -f mytunable
   ```

Files

/usr/sbin/tunrestore	Contains the **tunrestore** command.
/etc/tunables	Contains tunable files.
/etc/tunables/nextboot	Contains the values to be applied during the next boot.
/etc/tunables/lastboot	Contains the values of all tunables after the last boot.
/etc/tunables/lastboot.log	Contains messages, warnings and errors emitted by tunrestore during the last boot.

Related Information

The **raso**, **schedo**, **vmo**, **ioo**, **no**, and **nfso** commands.

The "tunchange Command" on page 526, "tuncheck Command" on page 527, "tundefault Command" on page 528, and "tunsave Command."

Tunables File Format in *AIX 5L Version 5.3 Files Reference*.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*.

tunsave Command

Purpose

Saves current tunable parameter values to a file.

Syntax

tunsave [**-a** | **-A**] **-f** | **-F** *Filename* [**-d** *Description*]

Description

The **tunsave** command saves the current state of tunable parameters in a file.

If *Filename* does not already exist, a new file is created. If it already exists, an error message prints unless the **-F** flag is specified, in which case, the existing file is overwritten.

Flags

-a	Saves all tunable parameters, including those who are currently set to their default value. These parameters are saved with the special value DEFAULT.
-A	Saves all tunable parameters, including those who are currently set to their default value. These parameters are saved numerically, and a comment, # DEFAULT VALUE, is appended to the line to flag them.
-d *Description*	Specifies the text to use for the *Description* field. Special characters must be escaped or quoted inside the *Description* field.
-f *Filename*	Specifies the name of the tunable file where the tunable parameters are saved. If *Filename* already exists, an error message prints. If it does not contain the '**/**' (forward slash) character, the *Filename* is relative to **/etc/tunables**.
-F *Filename*	Specifies the name of the tunable file where the tunable parameters are saved. If *Filename* already exists, the existing file is overwritten. If it does not contain the '**/**' (forward slash) character, the *Filename* is relative to **/etc/tunables**.

Examples

1. To save all tunables different from their default value into **/etc/tunables/mytunable**, type:

   ```
   tunsave -f mytunable
   ```

2. To save all tunables, including those who are currently set to their default value, but replace the default values with the special value DEFAULT, type:

   ```
   tunsave -a -f /home/admin/mytunable
   ```

3. To save all tunables, including those who are currently set to their default value using all numerical values, but flag the default vaules with the comment DEFAULT VALUE, type:

   ```
   tunsave -A -f mytunable
   ```

Files

/usr/bin/tunsave	Contains the tunsave command.
/etc/tunables	Contains all the saved files.

Related Information

The **raso**, **schedo**, **vmo**, **ioo**, **no**, and **nfso** commands.

The “tunchange Command” on page 526, “tuncheck Command” on page 527, “tundefault Command” on page 528, and “tunrestore Command” on page 529.

Tunables File Format in *AIX 5L Version 5.3 Files Reference*.

Kernel Tuning in *AIX 5L Version 5.3 Performance Tools Guide and Reference*.

---

turnacct Command

Purpose

Provides an interface to the **accton** command to turn process accounting on or off.

Syntax

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

Description

The **turnacct** command provides an interface to the **accton** command to turn process accounting on or off. You must specify whether you want process accounting on or off, because there is no default.

The **switch** flag turns off accounting and moves the current active data file (**/var/adm/pacct**) to the next free name in the **/var/adm/pacct***incr* file, where *incr* is a number starting at 1 and increased by one for each additional **pacct** file. After moving the **pacct** file, the **turnacct** command again turns on accounting.

The **turnacct switch** command is usually called by the **ckpacct** command, running under the **cron** daemon, to keep the active **pacct** data file a manageable size.

Security

Access Control: This command should grant execute (x) access only to members of the adm group.

Files

/usr/sbin/acct	Contains the path to the accounting commands.
/var/adm/pacct	Contains the current file for process accounting.
/var/adm/pacct*	Used if the **pacct** file gets too large.

Related Information

The **accton** command, **ckpacct** command.

The **cron** daemon.

Setting up an accounting subsystem in *Operating system and device management* explains the steps you must take to establish an accounting system.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the System accounting in *Operating system and device management*.

---

turnoff Command

Purpose

Sets the permission codes off for files in the **/usr/games** directory.

Syntax

turnoff

Description

The **turnoff** command sets the permission codes of files in the **/usr/games** directory. Root user authority is required to run this command.

The **turnoff** command looks for files in **/usr/games** whose permissions are set to 111 and sets these permissions to 000. If you install any new games in the **/usr/games** directory, set these permissions to 111.

Files

/usr/games	Contains the location of the system's games.

Related Information

The **arithmetic** command, **back** command, **bj** command, **craps** command, **fish** command, **fortune** command, **hangman** command, **moo** command, **number** command, **quiz** command, **ttt** command, **turnon** command, **wump** command.

---

turnon Command

Purpose

Sets permission codes on for files in the games directory.

Syntax

turnon

Description

The **turnon** command sets the permission codes of files in the **/usr/games** directory. Root user authority is required to run this command.

The **turnon** command looks for files with permissions set to 000 and sets them to 111 (execute permission for all users). If you install any new games in the **/usr/games** directory, set these permissions to 111.

File

/usr/games Contains the location of the system's games.

Related Information

The **arithmetic** command, **back** command, **bj** command, **craps** command, **fish** command, **fortune** command, **hangman** command, **moo** command, **number** command, **quiz** command, **ttt** command, **turnoff** command, **wump** command.

---

tvi Command

Purpose

Provides a trusted editor with a full screen display.

Syntax

tvi [**- l**] [**-R**] [**-w** *Number*] [**-c** [*Subcommand*]] [*File* ...]

Description

The **tvi** command calls the **tvi** editor, a trusted version of the **vi** editor, to edit the file or files specified by the *File* parameter. Files are edited in the order specified. If you do not provide a file name, the command opens a new file in which you can create text, but if you try to save the text to a file, you are prompted to add a file name to the save command, such as **:w** *File*. See the Examples section for more information.

You enter and leave the **tvi** editor in command mode, but to add or change text, you must enter text input mode. See the description of text input mode for information about the subcommands that initiate text input mode. To leave text input mode, press the **Esc** key. This returns you to command mode where you can save the text to a file with one of the **:w** commands, and exit the **tvi** editor, for example, with the **:q** command.

Because the full-screen display editor started by the **tvi** command is based on the **ex** editor, you can use the **ex** subcommands within the **tvi** editor. Subcommands function at the cursor position on the display screen.

The **tvi** editor makes a copy of the file you are editing in an edit buffer. The contents of the file are not changed until you save the changes.

Note: Several functions of the **vi** editor are not supported by the **tvi** editor. If you refer to information on the **vi** editor, be aware that the **-r** flag, the **-t** flag, shell escapes, user-defined macros, key mapping, and setting **vi** options permanently are not supported by the **tvi** editor.

tvi Editor Limitations

The maximum limits of the **tvi** editor assume single-byte characters. The limits are as follows:

- 256 characters per global command list
- 2048 characters in a shell escape command

- 128 characters in a string-valued option
- 30 characters in a tag name
- 524,230 lines silently enforced
- 128 map macros with 2048 characters total

Editing Modes

The **tvi** editor operates in the following modes:

command mode
The **tvi** editor starts in command mode. Any subcommand can be called except those that only correct text during text input mode. To see a description of the subcommands, refer to the topics in Subcommands for the tvi Editor. To identify the subcommands that cannot be called from command mode, refer to Changing Text While in Input Mode. The **tvi** editor returns to command mode when subcommands and other modes end. Press the Esc key to cancel a partial subcommand.

text input mode
The **tvi** editor enters text input mode when you use a permitted command that adds or changes text. To see a list of subcommands that initiate text input mode, refer to Adding Text to a File and the subcommands that change text from command mode, the **C** subcommand and the **c*x*** subcommands. After entering one of these subcommands, you can edit text with any of the subcommands that function in text input mode. To see a description of the subcommands, refer to the topics in "Subcommands for the tvi Editor". To return to command mode from text input mode, press the Esc key for a typical exit or press the Ctrl+C keys to create an **INTERRUPT** signal.

last line mode
Some subcommands read input on a line displayed at the bottom of the screen. These subcommands include those with the prefix : (colon), / (slash), and ? (question mark). When you enter the initial character, the **tvi** editor places the cursor at the bottom of the screen so you can enter the remaining command characters. To run the subcommand, press the Enter key. To cancel the subcommand, press the Ctrl+C keys to create an **INTERRUPT** signal. When you use the : (colon) to enter last line mode, the following characters have special meaning when used before commands that specify counts:

%	All lines regardless of cursor position
$	Last line
.	Current line

Customizing the tvi Editor

You can customize the **tvi** editor on a temporary basis by following the directions in ″Setting vi Editor Options″. The section on ″Setting vi Options Permanently″ is *not* applicable to the **tvi** editor.

Subcommands for the tvi Editor

Information on **vi** editor subcommands that are applicable to the **tvi** editor is summarized in the following list:

- vi General Subcommand Syntax.
- vi Subcommands for Adjusting the Screen.
- Editing Text with the **vi** Editor.
- Entering Shell Commands within the **vi** Editor is *not* supported by the **tvi** editor.
- Manipulating Files with the **vi** Editor.
- Subcommands for Interrupting and Ending the **vi** Editor.

Flags

-c [*Subcommand***]**
Carries out the **ex** editor subcommand before editing begins. This provides a line-oriented text editor. When a null operand is entered for the *Subcommand* parameter, as in `-c ''`, the editor places the cursor on the last line of the file.

-l	Enters the editor in LISP mode. In this mode, the editor indents appropriately for LISP code, and the **(**, **)**, **{**, **}**, **[[**, and **]]** subcommands are modified to act appropriately for LISP. These subcommands place the cursor at the specified LISP function. For more information on the LISP subcommands, refer to Moving to Sentences, Paragraphs, and Sections.
-R	Sets the **readonly** option to protect the file against overwriting.
-w *Number*	Sets the default window size to the value specified by the *Number* parameter. This is useful when you use the editor over a low-speed line.
+ [*Subcommand***]**	Same as the **-c** Subcommand.

Security

Access Control: This command should grant execute (x) access to all users and have the **trusted computing base** attribute.

Auditing Events:

Event	Information
TVI	filename

Examples

1. To call a trusted editor to edit the `plans` file, type:

   ```
   tvi plans
   ```

   This command puts the **tvi** editor into command mode. To add or change text, you must enter text input mode or use a command accepted in command mode. For more information, refer to the description of text input mode.
2. To save the text you create with the **tvi** editor, leave text input mode by pressing the Esc key, and then enter one of the save commands **:w**, **:w** *File*, or **:w!** *File*, for example:

   ```
   :w plans
   ```

   In this example, a file name, such as `plans`, is needed if you gave the **tvi** command without specifying a file name. If the file is already named, the **:w** command would not need the *File* parameter. If you want to overwrite an existing file, use the **:w!** *File* command, specifying the file you want to overwrite with the *File* parameter.

   If you try to save an unnamed file without supplying a file name, the following message appears:

   ```
   No current filename
   ```

   If this happens, repeat the **:w** command with a file name.
3. To exit the **tvi** editor from text input mode, press the Esc key to type command mode, and then type:

   ```
   :q!
   ```

   If the editor already is in command mode, you do not need to press the Esc key before giving the quit (**q!**) command.

Files

/usr/bin/tvi	Contains the **tvi** command.

Related Information

The **ex** command, **vi** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the Securing the network in *Security*.

twconvdict Command

Purpose

Converts other user dictionary to the operating system user dictionary. This command only applies to AIX 4.2 or higher.

Syntax

twconvdict [**-i** *Type*] [**-v** *CodePage*] [**-f** *Source*] [**-t** *Target*]

Description

The **twconvdict** command converts a dictionary to an operating system user dictionary. The supported code pages are SOPS, PS55 and ET. The dictionary type include both Tseng_Jye and Phonetic user dictionaries.

Flags

-f *Source* Specifies the name of font file to convert.

-i *Type* Specifies the type of dictionary to convert to. *Type* can be:

TJ Tseng_Jye, or

PH Phonetic.

-t *Target* Specifies the name of the converted font file.

-v *CodePage* Specifies the type of code page to convert to. *CodePage* can be:

SOPS

PS55, or

ET.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

To convert the dictionary `USRFONT.C12` to an operating system dictionary of code page of type SOPS and dictionary type of Tseng_Jye with the name `aix`, enter:

```
twconvdict -i TJ -v SOPS -f USRFONT.C12 -t aix
```

Files

/usr/lpp/tls/bin/twconvdict	Contains the **twconvdict** command.

---

twconvfont Command

Purpose

Converts other font files to a BDF font file.

Syntax

twconvfont [**-v** *CodePage*] [**-f** *Source*] [**-t** *Target*]

Description

The twconvfont command converts one font file type to the BDF font file. The supported code pages are SOPS, PS55 and ET.

Flags

-f *Source*	Specifies the name of font file to convert.
-t *Target*	Specifies the name of the converted font file.
-v *CodePage*	Specifies the type of code page to convert to. *CodePage* can be: **SOPS** **PS55**, or **ET**.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Examples

To convert the font file `USRFONT.C12` to a BDF font file of code page of type SOPS with the name `user.bdf`, enter:

```
twconvfont -v SOPS -f USRFONT.C12 -t user.bdf
```

Files

/usr/lpp/tls/bin/twconvfont	Contains the **twconvfont** command.

type Command

Purpose

Writes a description of the command type.

Syntax

type *CommandName* ...

Description

The standard output of the **type** command contains information about the specified command and identifies whether this is a shell built-in command, subroutine, alias, or keyword. The **type** command indicates how the specified command would be interpreted if used. Where applicable, the **type** command displays the related path name.

Because the **type** command must know the contents of the current shell environment, it is provided as a Korn shell or POSIX shell regular built-in command. If the **type** command is called in a separate command execution environment, the command may not produce accurate results. This would be the case in the following examples:

```
nohup type writer
find . -type f | xargs type
```

Exit Status

The following exit values are returned:

0 Successful completion.
>0 An error occurred.

Examples

1. To learn whether the **cd** command is a base command or an alias or some other command type, enter:

   ```
   type cd
   ```

   The screen displays the following information:

   ```
   cd is a shell builtin
   ```

2. To see the location of the **find** command, enter:

   ```
   type find
   ```

   The screen displays the following information:

   ```
   find is /usr/bin/find
   ```

Files

/usr/bin/ksh Contains the Korn shell **type** built-in command.

Related Information

The **bsh** command, **command** command, the **ksh** command.

ucfgif Method

Purpose
Unloads an interface instance from the kernel.

Syntax
ucfgif [**-l** *InterfaceInstance*]

Description
The **ucfgif** method removes an interface instance from the kernel. To remove the interface instance, the **ucfgif** method does the following:

1. Unloads the interface software by calling the **/usr/sbin/ifconfig** interface detach.
2. Sets the status flag of the interface instance to **defined**.

> **Note:** The **ucfgif** method is a programming tool and should not be executed from the command line.

Flags

-l *InterfaceInstance*	Specifies the interface instance to be unconfigured. If no interface name is specified, all configured interface instances are unconfigured.

Example
To remove an interface instance from the kernel, enter the method in the following format:

```
ucfgif  -l tr0
```

In this example, the name of the interface instance is `tr0`.

Related Information
The **ifconfig** command.

The **odm_run_method** subroutine.

TCP/IP network interfaces in *Networks and communication management*.

Writing a Device Method in *Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *General Programming Concepts*.

ucfginet Method

Purpose
Unloads the Internet instance and all related interface instances from the kernel.

Syntax
ucfginet

Description

The **ucfginet** method unloads the Internet instance from the kernel. This subroutine also deletes the appropriate entries in the Address Family Domain switch table and in the Network Input Interface switch table. The **ucfginet** method also sets the status flag of the instance to **defined**. The **ucfginet** method is called by the **rmdev** high-level command.

> **Note:** The **ucfginet** method is a programming tool and should not be executed from the command line.

Related Information

The **cfginet** method, **rmdev** command, **ucfgif** method.

The **odm_run_method** subroutine.

TCP/IP network interfaces in *Networks and communication management*.

Writing a Device Method in *Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *General Programming Concepts*.

---

ucfgqos Method

Purpose

Unconfigures and unloads the Quality of Service (QoS) instance from the kernel.

Syntax

ucfgqos

Description

The **ucfgqos** method disables Quality of Service (QoS) for the TCP/IP protocol suite on a host. This method detaches the QoS instance from the TCP/IP instance and unloads it from the kernel.

> **Note:** The **ucfgqos** method is a programming tool and is not intended to be invoked from the command line.

Example

To configure QoS on a host, use the following format:

```
ucfgqos
```

Related Information

The **cfgqos** method, and **ucfginet** method.

TCP/IP quality of service (QoS) in the *Networks and communication management*.

---

ucfgvsd Command

Purpose

ucfgvsd – Unconfigures a virtual shared disk.

Syntax

ucfgvsd {**–a** | *vsd_name* ...}

Description

The **ucfgvsd** command unconfigures the specified virtual shared disks. The specified virtual shared disks must be in the stopped state to be unconfigured. This command does not change any virtual shared disk definitions. It moves virtual shared disks from the stopped state to the defined state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Unconfigure a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Flags

–a Specifies that all virtual shared disks in the stopped state are to be unconfigured.

Parameters

vsd_name Specifies a virtual shared disk. The disk specified must be in the stopped state. If all disks have been unconfigured, and you specify VSD0, this command will attempt to unload the device driver from the kernel.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

Under normal circumstances, you should not issue this command. The Recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Standard Output

Current RVSD subsystem run level.

Examples

To unconfigure the virtual shared disk **vsd1vg1n1** in the stopped state, enter:

```
ucfgvsd vsd1vg1n1
```

Location

/opt/rsct/vsd/bin/ucfgvsd

Related Information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startvsd**, **stopvsd**, **suspendvsd**

uconvdef Command

Purpose

Compiles or generates a UCS-2 (Unicode) conversion table for use by the **iconv** library.

Syntax

uconvdef [**-f** *SrcFile*] [**-v**] *UconvTable*

Description

The **uconvdef** command reads *SrcFile* and creates a compiled conversion table in *UconvTable*. The *SrcFile* defines a mapping between UCS-2 and multibyte code sets (one or more bytes per character). The *UconvTable* is in a format that can be loaded by the UCSTBL conversion method located in the **/usr/lib/nls/loc/uconv** directory. This method uses the table to support UCS-2 conversions in both directions.

Flags

-f *SrcFile*	Specifies the conversion table source file. If this flag is not used, standard input is read.
-v	Causes output of the processed file statements.
UconvTable	Specifies the path name of the compiled table created by the **uconvdef** command. This should be the name of the code set that defines conversions into and out of UCS-2.

Exit Status

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

Examples

To access the compiled UCS-2 conversion table:

1. Create the compiled *UconvTable* using the name of the multibyte code set. For example, the conversion table between IBM-850 and UCS-2 can be compiled as follows:

   ```
   uconvdef -f IBM-850.ucmap IBM-850
   ```

2. Place the table in a directory called **uconvTable**. The default system directory is **/usr/lib/nls/loc/uconvTable**. If another directory is used, the **LOCPATH** environment variable needs to be set to include the parent directory (for example, **/usr/lib/nls/loc**).

   ```
   mv IBM-850 /usr/lib/nls/loc/uconvTable
   ```

3. Create symbolic links for conversions in each direction in a directory called **iconv**. The names for these links should be formed by concatenating the ″From″ code set and the ″To″ code set, separated by an underscore. The links should be set to point to the **/usr/lib/nls/loc/uconv/UCSTBL** conversion method. The default directory for these links is **/usr/lib/nls/loc/iconv**. If another directory is used, the **LOCPATH** environment variable needs to be set to include the parent directory (for example, **/usr/lib/nls/loc**).

   ```
   ln -s /usr/lib/nls/loc/uconv/UCSTBL \
   /usr/lib/nls/loc/iconv/IBM-850_UCS-2
   ln -s /usr/lib/nls/loc/uconv/UCSTBL \
   /usr/lib/nls/loc/iconv/UCS-2_IBM-850
   ```

   Note: The \ (backslash) is a line continuation character that is only needed if the command is broken into two lines.

Related Information

The **iconv** command.

The **iconv** subroutine, **iconv_close** subroutine, **iconv_open** subroutine.

The **unconvdef** source file format.

Code Set Overview, Converters Overview, and List of UCS-2 Interchange Converters in *AIX 5L Version 5.3 National Language Support Guide and Reference*.

---

udefif Method

Purpose

Removes an interface object from the system configuration database.

Syntax

udefif [**-l** *InterfaceInstance*]

Description

The **udefif** method deletes the specified interface instance from the system configuration database by:

1. Removing the database object associated with the interface instance.
2. Removing the connection and attribute information associated with the interface instance.

Flags

-l *InterfaceInstance*	Specifies the interface instance to be undefined. If no interface instances are specified, the **udefif** method undefines all defined interface instances.

Example

To remove an interface instance from the database, enter a method similar to the following:

```
udefif -l tr0
```

In this example, the interface instance to be removed is `tr0`.

Related Information

The **rmdev** command, **udefinet** method.

The **odm_run_method** subroutine.

TCP/IP network interfaces in *Networks and communication management*.

Writing a Device Method in *Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *General Programming Concepts*.

---

udefinet Method

Purpose

Undefines the Internet instance in the configuration database.

Syntax

udefinet

Description

The **udefinet** method removes the database information associated with the Internet instance, including attribute information associated with the Internet instance.

> **Note:** The **udefinet** method is a programming tool and should not be executed from the command line.

Related Information

The **rmdev** command.

The **odm_run_method** subroutine.

TCP/IP network interfaces in *Networks and communication management*.

Writing a Device Method in *Kernel Extensions and Device Support Programming Concepts*.

Object Data Manager (ODM) Overview for Programmers in *General Programming Concepts*.

---

udfcheck Command

Purpose

Performs a file system check on a UDF file system.

Syntax

udfcheck -d *device* [**-t** *tempfile*]

Description

The **udfcheck** command checks and repairs the UDF volume on a specified device.

Flags

-d *device*	Specifies the device on which **udfcheck** checks and repairs the UDF volume.
-t *tempfile*	Specifies a file where the **udfcheck** command stores information needed to perform a file system check.

Examples

1. To check the content of the UDF file system on device **/dev/cd1**, enter the following:

```
udfcheck -d /dev/cd1
```

Files

/usr/sbin/udfcheck	Contains the **udfcheck** command
/usr/lib/libudf.a	Contains the library routines called by the **udfcheck** command

Related Information

"udfcreate Command" and "udflabel Command."

---

udfcreate Command

Purpose

Creates UDF file systems.

Syntax

udfcreate -d *device* [**-b** *bitmap_location*] [**-f** *formatType*]

Description

The **udfcreate** command creates a UDF file system on the specified device and labels it with the generic set ID (*setID*) and volume name (*volName*).

Flags

-b	Specifies the *bitmap_location*. It can be one of the following, **b**, **e**, or **m**. **b** indicates that the location of the bitmap will be placed at the beginning of the partition. **e** indicates that the location of the bitmap will be placed at the end of the partition. **m** indicates that the location of the bitmap will be placed at the middle of the partition. The default location of the bitmap is the beginning of the partition.
-d *device*	Specifies the device on which to create the UDF volume.
-f	Indicates which version of UDF you want to be on the media. Format type of 1 represents UDF 1.5 version, 2 represents UDF 2.0 version, and 3 represents UDF 2.01 version. The default version is UDF 1.5.

Examples

1. To create a new UDF file system on device **/dev/cd1**, enter the following:

```
udfcreate -d /dev/cd1
```

Files

/usr/sbin/udfcreate	Contains the **udfcreate** command
/usr/lib/libudf.a	Contains the library routines called by the **udfcreate** command

Related Information

"udflabel Command" and "udfcheck Command" on page 545.

---

udflabel Command

Purpose

Fetches and changes the label on a UDF file system.

Syntax

udflabel -d *device* [**-l** *label*]

Description

The **udflabel** command displays or changes a UDF volume name. If there is no label provided, it displays the current UDF volume name on the device. If there is a label provided, it sets the current UDF volume name on the device to the new label.

Flags

-d *device*	Specifies the device containing the UDF volume.
-l *label*	Sets the label on the current UDF volume.

Examples

1. To change the current label on device **/dev/cd1 hello**, enter the following:
   ```
   udflabel -d /dev/cd1 -l hello
   ```
2. To display the current label on **device /dev/cd1**, enter the following:
   ```
   udflabel -d /dev/cd1
   ```

Files

/usr/sbin/udflabel	Contains the **udflabel** command
/usr/lib/libudf.a	Contains the library routines called by the **udflabel** command

Related Information

"udfcreate Command" on page 546 and "udfcheck Command" on page 545.

---

uil Command

Purpose

Starts the User Interface Language (UIL) compiler for the AIXwindows system.

Syntax

uil [**-I***PathName*] *InputFile* [**-m**] [**-o** *FileName*] [**-s**] [**-v** *FileName*] [**-w**] [**-wmd** *FileName*]

Description

The **uil** command calls the UIL compiler. The UIL is a specification language for describing the initial state of a user interface for an AIXwindows application. The specification describes the objects (menus, dialog boxes, labels, push buttons, and so on) used in the interface and specifies the functions to be called when the interface changes state as a result of user interaction.

Flags

-I*PathName*	Specifies **I**nclude*PathName* with no spaces. Causes the compiler to look for include files in a specified directory if include files were not found in the default paths. (uppercase i)

-m	Specifies that machine code is listed. This directs the compiler to place a description of the records that it added to the User Interface Definition (UID) in the listing file. This helps you isolate errors. The default is no machine code.
-o *FileName*	Directs the compiler to produce a UID. By default, UIL creates a UID with the name **a.uid**. The file specifies the file name for the UID. No UID is produced if the compiler issues any diagnostics categorized as error or severe.
-s	Directs the compiler to set the locale before compiling any files. The locale is set in an implementation-dependent manner. On ANSI C-based systems, the locale is usually set by calling the **setlocale (LC_ALL,** *″″***)** function. If this option is not specified, the compiler does not set the locale.
-v *FileName*	Directs the compiler to generate a listing. The file specifies the file name for the listing. If the **-v** option is not present, no listing is generated by the compiler. The default is no listing.
-w	Specifies that the compiler suppress all warning and informational messages. If this option is not present, all messages are generated, regardless of the severity.
-wmd *FileName*	Specifies a binary widget meta-language (WML) description file to be used instead of the default WML description.

Example

To start the UIL compiler, enter:

```
uil -I. -o ex.uid ex.uil
```

Exit Status

This command returns the following exit values:

0	Indicates successful completion.
>0	Indicates an error occurred.

Related Information

The **X** command.

---

uimx Command

Purpose

Starts the UIM/X user-interface management system for the X Window System.

Syntax

uimx [-dir *Path* **] [-file** *FileName*] **[-workspace** *Name* **] [-xrm** *Options* **]**

Description

The **uimx** command starts the UIM/X user-interface management system for the X Window System. It supports Motif 1.2 and provides a complete programming environment for developing graphical user interfaces (GUIs). UIM/X supports object-oriented programming in both C and C++.

UIM/X saves and loads text files that use the Xt resource syntax to describe interfaces and projects. It can also load UIL files. It generates C, C++, and UIL code. It can also generate makefiles, message catalogs, and resource files for an application.

UIM/X includes a built-in C Interpreter and the following tools and editors:

- Palette of Motif widgets
- Widget Browser for browsing complex widget hierarchies
- WYSIWYG layout editor for drawing interfaces

- Property Editor for setting initial values of widget properties; initial values can be literal values or C expressions
- Callback Editors for entering callback code
- Event, Action, and Translation Editors
- Menu and Main Window Editors
- Declarations Editor for editing the generated code for an interface
- Program Layout Editor for editing the generated main program and makefile; this editor gives you direct access to the main event loop

UIM/X supports two operating modes: Design and Test. In Test mode, the built-in C Interpreter allows you to test the behavior of your application. In Design mode, the C Interpreter validates the code you enter into the various UIM/X editors.

UIM/X provides a convenience library of functions that simplify the task of programming with X and Motif.

Flags

dir *Path*	Sets UIM/X's current directory to path.
file *FileName*	Loads an existing project, interface, or palette file called *FileName*. *FileName* can include an absolute path name, a path name relative to the current directory, or a path name relative to the **-dir** value.
workspace *Name*	Loads UIM/X into the corresponding CDE workspace called *name*.
xrm *Options*	Enables you to enter any resource specifications (*options*) that you would otherwise put in a resource file.

Security

Access Control: Any User

Files Accessed: None

Example

To start UIM/X, enter:

```
uimx
```

Files

/usr/uimx2.8/bin/uimx	Contains the **uimx** command.

ul Command

Purpose

Performs underlining.

Syntax

ul [**-i**] [**-t** *Terminal*] [*File* ...]

Description

The **ul** command reads the named files specified by the *File* parameter (or standard input if no file is given) and translates occurrences of underscores to the sequence that indicates underlining for the terminal in use, as specified by the **TERM** environment variable.

Flags

-i Causes the **ul** command to indicate underlining by a separate line containing appropriate _ (underline characters). Use this to see the underlining present in an **nroff** command output stream on a CRT terminal.

-t *Terminal* Overrides the terminal type specified in the environment. The **terminfo** file is read to determine the appropriate sequences for underlining. If the terminal is incapable of underlining, but is capable of a standout mode, then that mode is used instead. If the terminal can overstrike or automatically underline, the **ul** command acts like the **cat** command and displays on the screen. If the terminal cannot underline and no alternatives are available, underlining is ignored.

If the **-t** flag is not specified, the **ul** command translates for the terminal type specified by the **TERM** environment variable. If the value of the *Terminal* variable is not a valid terminal type, the **ul** command translates for a dumb terminal.

Files

/usr/share/lib/terminfo/* Contains the terminal capabilities database.

Related Information

The **cat** command, **colcrt** command, **man** command, **nroff** command.

The **terminfo** file.

---

ulimit Command

Purpose

Sets or reports user resource limits.

Syntax

ulimit [**-H**] [**-S**] [**-a**] [**-c**] [**-d**] [**-f**] [**-m**] [**-n**] [**-s**] [**-t**] [*Limit*]

Description

The **ulimit** command sets or reports user process resource limits, as defined in the **/etc/security/limits** file. This file contains these default limits:

```
fsize = 2097151
core = 2097151
cpu = -1
data = 262144
rss = 65536
stack = 65536
nofiles = 2000
```

These values are used as default settings when a new user is added to the system. The values are set with the **mkuser** command when the user is added to the system, or changed with the **chuser** command.

Limits are categorized as either soft or hard. With the **ulimit** command, you can change your soft limits, up to the maximum set by the hard limits. You must have root user authority to change resource hard limits.

Many systems do not contain one or more of these limits. The limit for a specified resource is set when the *Limit* parameter is specified. The value of the *Limit* parameter can be a number in the unit specified with each resource, or the value `unlimited`. To set the specific ulimit to `unlimited`, use the word `unlimited`

Note: Setting the default limits in the **/etc/security/limits** file sets system wide limits, not just limits taken on by a user when that user is created.

The current resource limit is printed when you omit the *Limit* parameter. The soft limit is printed unless you specify the **-H** flag. When you specify more than one resource, the limit name and unit is printed before the value. If no option is given, the **-f** flag is assumed.

Since the **ulimit** command affects the current shell environment, it is provided as a shell regular built-in command. If this command is called in a separate command execution environment, it does not affect the file size limit of the caller's environment. This would be the case in the following examples:

```
nohup ulimit -f 10000
env ulimit 10000
```

Once a hard limit has been decreased by a process, it cannot be increased without root privilege, even to revert to the original limit.

For more information about user and system resource limits, refer to the **getrlimit**, **setrlimit**, or **vlimit** subroutine in *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 1*.

Flags

-a Lists all of the current resource limits.
-c Specifies the size of core dumps, in number of 512-byte blocks.
-d Specifies the size of the data area, in number of K bytes.
-f Sets the file size limit in blocks when the *Limit* parameter is used, or reports the file size limit if no parameter is specified. The **-f** flag is the default.
-H Specifies that the hard limit for the given resource is set. If you have root user authority, you can increase the hard limit. Anyone can decrease it.
-m Specifies the size of physical memory, in number of K bytes.
-n Specifies the limit on the number of file descriptors a process may have.
-s Specifies the stack size, in number of K bytes.
-S Specifies that the soft limit for the given resource is set. A soft limit can be increased up to the value of the hard limit. If neither the **-H** nor **-S** flags are specified, the limit applies to both.
-t Specifies the number of seconds to be used by each process.

Exit Status

The following exit values are returned:

0 Successful completion.
>0 A request for a higher limit was rejected or an error occurred.

Example

To set the file size limit to 51,200 bytes, enter:

```
ulimit -f 100
```

Files

/usr/bin/ksh Contains the **ulimit** built-in command.

Related Information

The **ksh** command.

The **ulimit** subroutine, **getrlimit**, **setrlimit**, or **vlimit** subroutine in *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 1*.

---

umask Command

Purpose

Displays or sets the file mode creation mask.

Syntax

umask [**-S**] [*Mask*]

Description

If the *Mask* parameter is not specified, the **umask** command displays to standard output the file mode creation mask of the current shell environment. If you specify the *Mask* parameter using a three-digit octal number or symbolic code, the **umask** command sets the file creation mask of the current shell execution environment. The bits set in the file creation mask are used to clear the corresponding bits requested by an application or command when creating a file.

The **chmod** command describes how to use the symbolic and numeric codes to set permissions.

The **-S** flag produces symbolic output. If the flag is not specified, the default output format is octal.

If the **/usr/bin/umask** command is called in a subshell or separate command execution environment, it does not affect the file mode creation mask of the caller's environment. This would be the case in the following example:

```
(umask 002)
nohup umask ...
find . -exec umask ... \;
```

Flags

-S Produces symbolic output.

Exit Status

The following exit values are returned:

0 The file mode creation mask was successfully changed, or no *Mask* parameter was supplied.
>0 An error occurred.

Examples

1. To set the mode mask so that subsequently created files have their **S_IWOTH** bit cleared, enter either:

   ```
   umask a=rx,ug+w
   ```

   OR

   ```
   umask 002
   ```

   After setting the mode mask, display the current values of the mode mask by entering:

   ```
   umask
   ```

   The screen displays the following value:

```
02
```

2. To produce symbolic output, enter:

   ```
   umask -S
   ```

   The screen displays the following values:

   ```
   u=rwx,g=rwx,o=rx
   ```

3. Either numeric or symbol output can be used as the *Mask* parameter to a subsequent invocation of the **umask** command. Assume the mode mask is set as shown in example 2. To set the mode mask so that subsequently created files have their **S_IWGRP** and **S_IWOTH** bits cleared, enter:

   ```
   umask g-w
   ```

4. To set the mode mask so that subsequently created files have all their write bits cleared, enter:

   ```
   umask -- -w
   ```

   Note: The **-r**, **-w**, and **-x** *Mask* parameter values (or anything beginning with a hyphen) must be preceded by — (double hyphen, no space between) to keep it from being interpreted as an option.

Files

/usr/bin/ksh	Contains the Korn shell **umask** built-in command.
/usr/bin/umask	Contains the **umask** command.

Related Information

The **bsh** command, **chmod** command, **csh** command, **ksh** command.

---

umcode_latest Command

Purpose

Identifies system resources with firmware or microcode that can be updated from a specified source of image files.

Syntax

umcode_latest [**-s** *source*] [**-l**] [**-A**] | [**-a**[**-q**][**-r**]| **-i**] | **-h**

Description

The **umcode_latest** command lists or downloads the system resources that have an older firmware or microcode level than the firmware or microcode level that was found on the specified source for those system resources.

Note: System Firmware images of system types 8842/8844/7047/7013/7015/7017 and 7025-F50 are not supported by this command. For systems with temporary and permanent system firmware images, the **umcode_latest** command uses the temporary system firmware image for comparisons with the images on the specified source. System firmware image file names must end with **.img**.

Flags

-a	Updates all system resources that have newer microcode on the source.

-A	Lists or updates resource when any of the images on the source is different from the image currently listed or updated. The default is to list or update whenever the source has a newer image.
-h	Provides extended usage help.
-i	Provides an interactive mode so that each resource that needs an update is prompted.
-l	Lists the system resources that need updates. This is the default.
-q	Refrains from asking whether to proceed with the update all.
-r	Refrains from asking whether to proceed with the update requiring a system IPL.
-s *source*	Points to the source of the microcode image. The default is **/etc/microcode**.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Examples

1. To list all system resources with firmware or microcode that can be updated from the images in **/etc/microcode**, enter:
   ```
   /usr/lpp/diagnostics/bin/umcode_latest
   ```
2. To list all system resources with firmware or microcode that can be updated from the images that are in the **/tmp/fwupdate** directory, enter:
   ```
   /usr/lpp/diagnostics/bin/umcode_latest -s /tmp/fwupdate
   ```
3. To list all system resources with firmware or microcode that can be updated from the images that are in the **/tmp/fwupdate** directory, and for each resource ask whether the resource should be updated at this time, enter:
   ```
   /usr/lpp/diagnostics/bin/umcode_latest -s /fwupdate -i
   ```
4. To automatically update all of the system resources with firmware or microcode that have newer images on the ISO 9660 format CD-ROM, which has already been inserted into the cd1 drive, enter:
   ```
   /usr/lpp/diagnostics/bin/umcode_latest -s cd1 -a -q
   ```

Restrictions

System Firmware images of system types 8842/8844/7047/7013/7015/7017 and 7025-F50 are not supported by this command. For systems with temporary and permanent system firmware images, the **umcode_latest** command uses the temporary system firmware image for comparisons with the images on the specified source. System firmware image file names must end with **.img**.

Location

/usr/lpp/diagnostics/bin/umcode_latest

Related Information

The **diag** command.

umount or unmount Command

Purpose

Unmounts a previously mounted file system, directory, or file.

Syntax

{ **unmount** | **umount** } [**-f**] [**-a**] | [**all** | **allr** | *Device* | *Directory* | *File* | *FileSystem* | **-n** *Node* | **-t** *Type*]

Description

Another name for the **umount** command is the **unmount** command. Either name can be used. You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit umount
```

The **umount** command unmounts a previously mounted device, directory, file, or file system. Processing on the file system, directory, or file completes and it is unmounted. Members of the system group and users operating with root user authority can issue any **umount** command. Only users with root authority or are members of the system group can unmount a directory or file.

> **Note:** SMIT will not unmount the **/usr/lpp/info/$LANG** directory, the directory on which SMIT helps are located. Typically, this is the CD-ROM.

To unmount local mounts you can specify the device, directory, file, or file system on which it is mounted.

If the file system being unmounted is a JFS2 snapshot, the **umount** command will unmount the snapshot, though the snapshot will still be active. The **snapshot** command must be used to delete the snapshot.

If the file system being unmounted is a snapped file system with mounted snapshots, the **umount** command displays a warning that there are mounted snapshots and exits without unmounting the file system. The snapshots must be unmounted first.

Note: If the **cdromd** CD and DVD automount daemon is enabled, then those devices will be automatically mounted as specified in the **/etc/cdromd.conf** file. Use the **cdumount** or **cdeject** command to unmount an automounted CD or DVD. Use ″**stopsrc -s cdromd**″ to disable the CD/DVD automount daemon.

Flags

-a Unmounts all mounted file systems.

all Unmounts all mounted file systems.

allr Unmounts all remotely mounted file systems.

> **Note:** For remote mounts, specify the device, directory, file, or file system parameters. If you specify the **allr** flag, the **umount** command unmounts all remote mounts.

-f Forces an unmount in a remote environment. Use to free a client when the server is down and server path names cannot be resolved, or when a file system must be unmounted while it is still in use. The **-f** flag is not supported for journaled file systems.
Note: For remote file systems, using this flag causes all file operations on the file system except **close()** and **unmap()** to fail. Any file data that has been written by an application but has not yet transferred to the server will be lost. A forced unmount of an NFS version 4 file system can cause open file state for other file systems mounted from the same server to be lost as well.

-n *Node* Specifies the node holding the mounted directory you want to unmount. The **umount -n** *Node* command unmounts all remote mounts made from the *Node* parameter.

-t *Type* Unmounts all stanzas in the **/etc/filesystems** file that contain the **type=***Type* flag and are mounted. The *Type* parameter is a string value, such as the remote value that specifies the name of the group.

Note: You cannot use the **umount** command on a device in use. A device is in use if any file is open for any reason or if a user's current directory is on that device.

Examples

1. To unmount all mounts from remote node `Node A`, enter:

   ```
   umount  -n nodeA
   ```

2. To unmount files and directories of a specific type, enter:

   ```
   umount  -t test
   ```

   This unmounts all files or directories that have a stanza in the **/etc/filesystems** file that contains the **type=test** attribute.

Files

/etc/filesystems Lists the known file systems and defines their characteristics.

Related Information

The **cdcheck** command, **cdeject** command, **cdmount** command, **cdromd** command, **cdumount** command, **cdutil** command, **mount** command, **fuser** command.

The **mount** subroutine, **umount** subroutine, **vmount** subroutine, **uvmount** subroutine, **mntctl** subroutine.

The System Management Interface Tool (SMIT) in *Operating system and device management* explains the structure, main menus, and tasks that are done with SMIT.

The Mounting in *Operating system and device management* explains mounting files and directories, mount points, and automatic mounts.

---

umountall Command

Purpose

Unmounts groups of dismountable devices or filesystems.

Syntax

umountall [**-k**] [**-s**] [**-F** *FileSytemType*] [**-l** | **-r**]

umountall [**-k**] [**-s**] [**-h** *Host*]

Description

The **umountall** command by default unmounts all dismountable file systems or devices except root, /proc, /var and /usr. If the *FileSystemType* is specified, **umountall** limits its actions to the file system type specified. There is no guarantee that **umountall** will unmount busy file systems, even if the **-k** option is specified.

Flags

-F *FileSystemType*	Specifies the type of file systems to be dismounted. *FileSystemType* corresponds to the vfs column printed out by the mount command. All dismountable file systems of the given type will be unmounted. This flag cannot be used in combination with the **-h** flag.
-h *Host*	Specifies the host node. All file systems remotely mounted from this host will be unmounted.
-k	Sends a **SIGKILL** to each process on the mount point before unmounting. This option internally uses the **fuser -k** command to kill all the processes running on the mount point. As this option causes each process on the mount point to be killed, the unmount of the mount point does not happen immediately. There is no guarantee that **umountall** will unmount busy file systems, even if the **-k** option is specified. An attempt to unmount the mount point will be made only after all the processes using the mount point are killed.
-l	Limits the action to local filesystems.
-r	Limits the action to remote filesystems.
-s	This is a no-operation flag provided for System V compatibility on serializing the **unmounts**. The serialization of the **unmount** command is done using **-k** option by terminating all the associated processes on the mount point.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Examples

1. To unmount all dismountable file systems, enter:
   ```
   umountall
   ```
2. To unmount all dismountable filesystems of type **jfs**, enter:
   ```
   umountall -F jfs
   ```
3. To unmount all dismountable filesystems mounted from host.domain, enter:
   ```
   umountall -h host.domain
   ```
4. To unmount all remotely mounted filesystems, enter:
   ```
   umountall -r
   ```

Files

/usr/sbin/umountall	Contains the **umountall** command.

Related Information

The **umount** command.

unalias Command

Purpose
Removes alias definitions.

Syntax
unalias -a

unalias *AliasName* ...

Description
The **unalias** command removes the definition for each alias name specified, or removes all alias definitions if the **-a** flag is used. Alias definitions are removed from the current shell environment.

Since the **unalias** command affects the current shell execution environment, it is provided as a Korn shell or POSIX shell regular built-in command.

Flags
-a Removes all alias definitions from the current shell environment.

Exit Status
The following exit values are returned:

0 Successful completion.
>0 One of the alias names specified did not represent a valid alias definition, or an error occurred.

Files
/usr/bin/ksh	Contains the Korn shell **unalias** built-in command.
/usr/bin/unalias	Contains the **unalias** command.

Related Information
The **alias** command, **csh** command, **ksh** command.

uname Command

Purpose
Displays the name of the current operating system.

Syntax
uname [**-a** | **-x** | **-S** *Name*] [**-F**] [**-f**] [**-l**] [**-L**] [**-m**] [**-M**] [**-n**] [**-p**] [**-r**] [**-s**] [**-T** *Name*] [**-u**] [**-v**]

Description
The **uname** command writes to standard output the name of the operating system that you are using.

The machine ID number contains 12 characters in the following digit format: *xxyyyyyymmss*. The *xx* positions indicate the system and is always 00. The *yyyyyy* positions contain the unique ID number for the

entire system. The *mm* position represents the model ID. The *ss* position is the submodel number and is always 00. The model ID describes the ID of the CPU Planar, not the model of the System as a whole.

You can use the **uname -m** command sometimes to determine which model you are using. The following list is not complete. Refer to hardware vendor supplied documentation for values in the range E0 - FF. Also note that not all machine types have a machine ID. Many new machines share a common machine ID of 4C. Hexadecimal codes for the system models (*mm*) are:

Machine Type	Machine Model	Machine ID
7006	410	42
7007	N40	F0
7008	M20	43
7008	M20A	43
7009	C10	48
7011	220	41
7011	230	47
7011	250	46
7012	320	31
7012	320H	35
7012	340	37
7012	350	38 or 77
7012	355	77
7012	360	76
7012	365	76
7012	370	75
7012	375	75
7012	380	58
7012	390	57
7012	G30	A6
7012	G40	A7
7013	520	30
7013	520H	34
7013	530	10
7013	530H	18
7013	540	14 or 11
7013	550	1C
7013	550L	77
7013	560	5C
7013	570	67
7013	580	66
7013	58H	71
7013	590	70
7013	590H	72
7013	J30	A0
7013	J40	A1
7015	930	20 or 02
7015	950	2E
7015	970	63
7015	970B	63
7015	980	64
7015	980B	64
7015	990	80
7015	R10	67
7015	R20	72
7015	R24	81
7015	R30	A3
7015	R40	A4
7016	730	10
7018	740	30
7018	770	67
7024	E20	C0

7025	F30	C4
7030	3AT	58
7030	3BT	57
7043	140	4C
7043	240	4C
7248	43P	4C

The machine identifier value returned by the **uname** command may change when new operating system software levels are installed. This change affects applications using this value to access licensed programs. To view this identifier, enter the **uname -m** command.

Contact the appropriate support organization if your application is affected.

Flags

-a Displays all information specified with the **-m**, **-n**, **-r**, **-s**, and **-v** flags. Cannot be used with the **-x** or **-S***Name* flag. If the **-x** flag is specified with the **-a** flag, the **-x** flag overrides it.

-F Displays a system identification string comprised of hexadecimal characters. This identification string is the same for all partitions on a particular system.

-f Similar to the **F** flag, except that the partition number is also used in the calculation of this string. The resulting identification string is unique for each partition on a particular system.

-l Displays the LAN network number.

-L Displays LPAR number and LPAR name. If LPAR does not exist, -1 is displayed for LPAR number and **NULL** for LPAR name.

-m Displays the machine ID number of the hardware running the system.
Note: The **-m** flag cannot be used to generate a unique machine identifier for partitions in an LPAR environment.

-M Displays the system model name. If the model name attribute does not exist, a null string is displayed.

-n Displays the name of the node. This may be a name the system is known by to a UUCP communications network.

-p Displays the architecture of the system processor.

-r Displays the release number of the operating system.

-s Displays the system name. This flag is on by default.

-S *Name* Sets the name of the node. This can be the UUCP communications network name for the system.

-T *Name* Sets the system name. This can be the UUCP communications network name for the system.

-u Displays the system ID number. If this attribute is not defined, the output is the same as the output displayed by **uname -m**.

-v Displays the operating system version.

-x Displays the information specified with the **-a** flag as well as the LAN network number, as specified by the **-l** flag.

If you enter a flag that is not valid, the **uname** command exits with an error message, an error return status, and no output.

> **Note:** The uname command does not preserve the new system name and node name values across system reboot.

Exit Status

This command returns the following exit values:

0 The requested information was successfully written.

>0 An error occurred.

Example

To display the complete system name and version banner, enter:

```
uname -a
```

Files

/usr/bin/uname	Contains the **uname** command.

Related Information

The **uname** or **unamex** subroutine.

---

uncompress Command

Purpose

Restores compressed files.

Syntax

uncompress [**-c**] [**-F**] [**-f**] [**-n**] [**-q**] [**-V**] [*File ...*]

Description

The **uncompress** command restores original files that were compressed by the **compress** command. Each compressed file specified by the *File* parameter is removed and replaced by an expanded copy. The expanded file has the same name as the compressed version, but without the **.Z** extension. If the user has root authority, the expanded file retains the same owner, group, modes, and modification time as the original file. If the user does not have root authority, the file retains the same modes and modification time, but acquires a new owner and group. If no files are specified, standard input is expanded to standard output.

Flags

-c	Write to standard output. No files are changed.
-f or **-F**	Forces expansion. The **-f** and **-F** flags are interchangeable. Overwrites the file if it already exists. The system does not prompt the user that an existing file will be overwritten. File size may not actually shrink.
-n	Omits the compressed file header from the compressed file. **Note:** Use this option if the file was compressed using the **-n** flag. Otherwise, uncompressing the file will not work.
-q	Suppresses the display of compression statistics generated by the **-v** flag. If several **-v** and **-q** flags are on the same command line, the last one specified controls the display of the statistics.
-V	Writes the current version and compile options to standard error.

Parameters

File ...	Specifies the compressed files to restore.

Return Values

The **uncompress** command detects an error and exit with a status of 1 if any of the following events occur:

- The input file was not produced by the **compress** command.

- An input file cannot be read or an output file cannot be written.

If no error occurs, the exit status is 0.

Exit Status

0 Successful completion.
>0 An error occurred.

Example

To uncompress the `foo.Z` file, enter:

```
uncompress foo.Z
```

The `foo.Z` file is uncompressed and renamed `foo`.

Related Information

The **compress** command, **dmpuncompress** command, **pack** command, **unpack** command, **zcat** command.

Commands overview in *Operating system and device management*.

---

undefvsd Command

Purpose

undefvsd – Undefines a virtual shared disk.

Syntax

undefvsd *vsd_name* ...

Description

This command is used to remove a virtual shared disk definition and any special device files from **/dev** for the given *vsd_names* on all the virtual shared disk nodes. The virtual shared disks must be unconfigured and in the defined state on all the virtual shared disk nodes.

You can use the System Management Interface Tool (SMIT) to run the **undefvsd** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Undefine a Virtual Shared Disk** option.

Flags

None.

Parameters

vsd_name Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

To delete the information associated with the virtual shared disk **vsd1vg2n1**, enter:

```
undefvsd vsd1vg2n1
```

Location

/usr/lpp/vsd/bin/undefvsd

Related Information

Commands: **defvsd**

---

unexpand Command

Purpose

Writes to standard output with tabs restored.

Syntax

unexpand [**-a** | **-t** *TabList*] [*File ...*]

Description

The **unexpand** command puts tabs back into the data from the standard input or the named files and writes the result to standard output. By default, only leading spaces and tabs are reconverted to maximal strings of tabs.

Note: The *File* parameter must be a text file.

Flags

-a Inserts tabs wherever their presence compresses the resultant file by replacing two or more characters.

-t *TabList* Specifies the position of the tab stops. The default value of a tab stop is 8 column positions.

The *TabList* variable must consist of a single positive-decimal integer or multiple positive-decimal integers. The multiple integers must be in ascending order and must be separated by commas or by blank characters with quotation marks around the integers. The single *TabList* variable sets the tab stops an equal number of column positions apart. The multiple *TabList* variable sets the tab stop at column positions that correspond to the integers in the *TabList* variable.

A space-to-tab conversion does not occur for characters at positions beyond the last one specified in a multiple *TabList* variable.

Note: When the **-t** flag is specified, the **-a** flag is ignored and conversion is not limited to processing leading blank characters.

Exit Status

This command returns the following exit values:

0 The command ran successfully.
>0 An error occurred.

Example

To replace space characters with tab characters in the **xyz** file, enter:

```
unexpand xyz
```

Files

/usr/bin/unexpand Contains the **unexpand** command.

Related Information

The **expand** command, **newform** command, **tab** command, **untab** command.

Files in *Operating system and device management*.

Input and output redirection overview in *Operating system and device management*.

---

unfencevsd Command

Purpose

unfencevsd – Gives applications running on a node or group of nodes access to a virtual shared disk or group of virtual shared disks that were previously fenced from applications running on those nodes.

Syntax

unfencevsd {**–a** | **–v** *vsd_name_list*} {**–n** *node_list* [**–f**] }

Description

Under some circumstances, the system may believe a node has become inoperable and may begin recovery procedures when the node is actually operational, but is cut off from communication with other nodes running the same application. In this case, the problem node must not be allowed to serve requests for the virtual shared disks it normally manages until recovery is complete and the other nodes running the

application recognize the problem node as operational. The **fencevsd** command prevents the problem node from filling requests for its virtual shared disks. The **unfencevsd** command allows fenced nodes to regain access to the virtual shared disks.

You can issue this command from any node that is online in the peer domain.

Flags

–a Specifies all virtual shared disks.

–f Allows a fenced node to unfence itself.

–n *node_list* Specifies one or more node numbers separated by commas.

–v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To unfence node 5 from the virtual shared disks vsd1 and vsd2, enter:
   ```
   unfencevsd -v vsd1,vsd2 -n 5
   ```
2. To unfence node 7 from the virtual shared disks vsd1 and vsd2 when the **unfencevsd** command must be entered from node 7, enter:
   ```
   unfencevsd -v vsd1,vsd2 -n 7 -f
   ```

Location

/opt/rsct/vsd/bin/unfencevsd

Related Information

Commands: **fencevsd**, **lsvsd**

Refer to *RSCT: Managing Shared Disks* for more information on the Recoverable virtual shared disk subsystem and how you can use the **fencevsd**, **unfencevsd**, and **lsfensevsd** commands to preserve data integrity during application recovery.

unget Command (SCCS)

Purpose

Cancels a previous **get** command.

Syntax

unget [**-r***SID*] [**-s**] [**-n**] *File* ...

Description

The **unget** command allows you to restore a g-file created with **get -e** before the new delta is created. Any changes are therefore discarded. If you specify a **-** (dash) for the value of *File*, standard input is read, and each line of standard input is interpreted as the name of an SCCS file. An end-of-file character terminates input.

If you specify a directory for the *File* value, the **unget** command performs the requested actions on all SCCS files that are currently in the process of being edited (those files with the **s.** prefix).

Once you have run an **unget** command on a file, you must reissue a **get -e** command to make changes to the file. The **unget** command automatically deletes the g-file.

Flags

Each flag or group of flags applies independently to each named file.

-n Prevents the automatic deletion of the g-file. This flag allows you to retain the edited version of the file without making a delta.

-r*SID* Specifies the new delta that would have been created by the next use of the **delta** command. You must use this flag if you have two or more pending deltas to the file under the same login name. You can look at the p-file to see if you have more than one delta pending to a particular SID under the same login name. The *SID* specification must unambiguously specify only one SID to discard, or the **unget** command displays an error message and stops running.

-s Suppresses displaying the deleted SID.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Example

To discard the changes you have made to an SCCS file after running a **get -e** command, enter:

```
unget s.prog.c
```

Files

/usr/bin/unget Contains the path to the SCCS **unget** command.

Related Information

The **delta** command, **get** command, **sact** command.

The **sccsfile** file format.

List of SCCS Commands in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

Source Code Control System (SCCS) Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

unifdef Command

Purpose

Removes ifdef lines from a file.

Syntax

unifdef [**-t**] [**-l**] [**-c**] [**-D***Symbol*] [**-U***Symbo*l] [**-id***Symbol*] [**-iu***Symbol*] [*File*]

Description

The **unifdef** command is useful for removing ifdef lines from a file while otherwise leaving the file alone. The **unifdef** command recognizes nested ifdefs, comments, and single and double quotes of C syntax in order to function correctly, but does not include files or interpret macros. The **unifdef** command recognizes but does not remove comments.

The **unifdef** command takes its input from standard input if no *File* is specified and copies its output to standard output.

Once a *Symbol* is specified, the lines inside those ifdefs are copied to the output or removed, as appropriate. The ifdef, ifndef, else, elif, and endif lines associated with the symbol are also removed. Ifdefs that involve unspecified symbols are untouched and copied out along with their associated ifdef, else, elif, and endif lines. If the same symbol appears in more than one argument, only the first occurrence is significant. For instance, if an ifdef `X` occurs nested inside another ifdef `X`, the inside ifdef is considered an unrecognized symbol.

When using ifdefs to delimit non-C lines such as comments or unfinished code, it is necessary to specify which symbols are to be used for that purpose. Otherwise, the **unifdef** command will try to parse for quotes and comments in those ifdef lines.

The **unifdef** command cannot process **cpp** constructs such as:

```
#if defined(X) || defined(Y)
```

OR

```
#elif X
```

OR

```
#elif defined(X) || defined(Y)
```

Keywords

The following keywords are recognized by the **unifdef** command:

- **ifdef**
- **ifndef**
- **else**
- **endif**
- **elif**

Flags

-c	Complements the operation of the **unifdef** command. That is, the lines which would have been removed are retained and vice versa.
-D *Symbol*	Specifies the symbol to be defined.
File	Specifies the input source.
-id *Symbol*	The **unifdef** command will not try to recognize comments, single quotes, or double quotes inside specified **ifdefs**, but these lines will be copied out.
-iu *Symbol*	The **unifdef** command will not try to recognize comments, single quotes, or double quotes inside specified **ifdefs**. These lines will not be copied out.
-l	Causes removed lines to be replaced with blank lines instead of being deleted.
-t	Allows the **unifdef** command to be used for plain text (instead of C code): the **unifdef** command will not try to recognize comments, single quotes and double quotes.
-U *Symbol*	Specifies the symbol to be undefined.

Exit Status

This command returns the following exit values:

0	The output is an exact copy of the input.
1	The output is not an exact copy of the input.
2	The command failed due to a premature EOF, or to an inappropriate **else**, **elif**, or **endif**.

Examples

1. The following example:

   ```
   unifdef -DA original.c > modified.c
   ```

   causes the **unifdef** command to read the file `original.c`, and remove the #ifdef A lines. It then removes everything following an #elif/#else associated with the #ifdef A, down to the #endif. The output is placed in the `modified.c` file.
2. The following example:

   ```
   unifdef -UA original.c > modified.c
   ```

   causes the **unifdef** command to read the file `original.c`, and remove the #ifdef A down to either its associated #elif//#else, or its associated #endif. In the case of the #elif, the #elif is replaced with #if. In the case of #else, the #else is deleted along with its associated #endif. The output is placed in the `modified.c` file.

Files

/usr/bin/unifdef	Contains the **unifdef** command.

Related Information

The **cpp** command.

The Commands in *Operating system and device management*.

---

uniq Command

Purpose

Reports or deletes repeated lines in a file.

Syntax

uniq [**-c** | **-d** | **-u**] [**-f** *Fields*] [**-s** *Characters*] [**-***Fields*] [**+***Characters*] [*InFile* [*OutFile*]]

Description

The **uniq** command deletes repeated lines in a file. The **uniq** command reads either standard input or a file specified by the *InFile* parameter. The command first compares adjacent lines and then removes the second and succeeding duplications of a line. Duplicated lines must be adjacent. (Before issuing the **uniq** command, use the **sort** command to make all duplicate lines adjacent.) Finally, the **uniq** command writes the resultant unique lines either to standard output or to the file specified by the *OutFile* parameter. The *InFile* and *OutFile* parameters must specify different files.

The input file must be a text file. A *text* file is a file that contains characters organized into one or more lines. The lines can neither exceed 2048 bytes in length (including any newline characters) nor contain null characters.

The **uniq** command compares entire lines by default. If the **-f** *Fields* or **-***Fields* flag is specified, the **uniq** command ignores the number of fields specified by the *Fields* variable. A *field* is a string of characters separated from other character strings by one or more <blank> characters. If the **-s** *Characters* or **-***Characters* flag is specified, the **uniq** command ignores the number of characters specified by the *Characters* variable. Values specified for the *Fields* and *Characters* variables must be positive decimal integers.

The current national language environment determines the <blank> characters used by the **-f** flag as well as how the **-s** flag interprets bytes as a character.

The **uniq** command exits with a value of 0 if successful. Otherwise, it exits with a value greater than 0.

Flags

-c	Precedes each output line with a count of the number of times each line appeared in the input file.
-d	Displays only the repeated lines.
-f *Fields*	Ignores the number of fields specified by the *Fields* variable. If the value of the *Fields* variable exceeds the number of fields on a line of input, the **uniq** command uses a null string for comparison. This flag is equivalent to the **-***Fields* flag.
-u	Displays only the unrepeated lines.
-s *Characters*	Ignores the number of characters specified by the *Characters* variable. If the value of the *Characters* variable exceeds the number of characters on a line of input, the **uniq** command uses a null string for comparison. If both the **-f** and **-s** flags are specified, the **uniq** command ignores the number of characters specified by the **-s** *Characters* flag starting in the field following the fields specified by the **-f** *Fields* flag. This flag is equivalent to the **+***Characters* flag.
-*Fields*	Ignores the number of fields specified by the *Fields* variable. This flag is equivalent to the **-f** *Fields* flag.
+*Characters*	Ignores the number of characters specified by the *Characters* variable. If both the **-***Fields* and **+***Characters* flags are specified, the **uniq** command ignores the number of characters specified by the **+***Characters* flag starting in the field following the fields specified by the **-***Fields* flag. This flag is equivalent to the **-s** *Characters* flag.

Exit Status

This command returns the following exit values:

0	The command ran successfully.
>0	An error occurred.

Example

To delete repeated lines in a file named `fruit` and save it to a file named `newfruit`, enter:

```
uniq fruit newfruit
```

If the `fruit` file contains the following lines:

```
apples
apples
peaches
pears
bananas
cherries
cherries
```

then the `newfruit` file will contain the following lines after you run the **uniq** command:

```
apples
peaches
pears
bananas
cherries
```

Files

/usr/bin/uniq	Contains the **uniq** command.

Related Information

The **comm** command, **sort** command.

---

units Command

Purpose

Converts units in one measure to equivalent units in another measure.

Syntax

units [**-**] [*File*]

Description

The **units** command converts quantities expressed in one measurement to their equivalents in another. The **units** command is an interactive command. It prompts you for the unit you want to convert *from* and the unit you want to convert *to*. This command only does multiplicative scale changes. That is, it can convert from one value to another only when the conversion involves a multiplication. For example, it cannot convert between degrees Fahrenheit and degrees Celsius because the value of 32 must be added or subtracted in the conversion.

You can specify a quantity as a multiplicative combination of units, optionally preceded by a numeric multiplier.

Indicate powers by entering suffixed positive integers, and indicate division with a / (slash).

The **units** command recognizes `lb` as a unit of mass, but considers `pound` to be the British pound sterling. Compound names are run together (such as `lightyear`). Prefix British units differing from their American counterparts with `br` (`brgallon`, for instance).

The **/usr/share/lib/unittab** file contains a complete list of the units that the **units** command uses. You can also define new units in this file. The *File* parameter may be used to override the values of the standard conversion factors listed in the **/usr/share/lib/unittab** file. The specified file must follow the same format as the **unittab** file.

Most familiar units, abbreviations, and metric prefixes are recognized by the **units** command, as well as the following:

pi	Ratio of circumference to diameter
c	Speed of light
e	Charge on an electron
g	Acceleration of gravity
force	Same as **g**
mole	Avogadro's number
water	Pressure head per unit height of water
au	Astronomical unit

Flags

- Lists the conversion factors contained in the **/usr/share/lib/unittab** file before you are prompted to enter your conversion.

Examples

1. To display conversion factors for inches to centimeters, enter:

   ```
   units
   you have: in
   you want: cm
   ```

   The **units** command returns the following values:

   ```
   * 2.540000e+00
   / 3.937008e-01
   ```

   The output tells you to multiply the number of inches by `2.540000e+00` to get centimeters, and to multiply the number of centimeters by `3.937008e-01` to get inches.

   These numbers are in standard exponential notation, so `3.937008e-01` means `3.937008 x 10-1`, which is the same as `0.3937008`.

   > **Note:** The second number is always the reciprocal of the first; for example, `2.54` equals `1/0.3937008`.

2. To convert a measurement to different units, enter:

   ```
   units
   you have: 5 years
   you want: microsec
   ```

   The **units** command returns the following values:

   ```
   * 1.577846e+14
   / 6.337753e-15
   ```

   The output shows that `5 years` equals `1.577846 x 1014` microseconds, and that one microsecond equals `6.337753 x 10-15` years.

3. To give fractions in measurements, enter:

   ```
   units
   you have: 1|3 mi
   you want: km
   ```

The **units** command returns the following values:

```
* 5.364480e-01
/ 1.864114e+00
```

The | (vertical bar) indicates division, so `1|3` means one-third. This shows that one-third mile is the same as `0.536448` kilometers.

4. To include exponents in measurements, enter:

   ```
   units
   you have: 1.2-5 gal
   you want: floz
   ```

   The **units** command returns the following values:

   ```
   * 1.536000e-03
   / 6.510417e+02
   ```

   The expression `1.2-5 gal` is the equivalent of `1.2 x 10-5`. Do *not* type an `e` before the exponent (that is, `1.2e-5 gal` is not valid). This example shows that `1.2 x 10-5` (`0.000012`) gallons equal `1.536 x 10-3` (`0.001536`) fluid ounces.

5. To specify complex units, enter:

   ```
   units
   you have: gram centimeter/second2
   you want: kg-m/sec2
   ```

   The **units** command returns the following values:

   ```
   * 1.000000e-05
   / 1.000000e+05
   ```

   The units `gram centimeter/second2` mean ″grams x centimeters/second2.″ Similarly, `kg-m/sec2` means ″kilograms x meters/sec2,″ which is often read as ″kilogram-meters per seconds squared.″

6. If the units you specify after `you have:` and `you want:` are incompatible:

   ```
   you have: ft
   you want: lb
   ```

   The **units** command returns the following message and values:

   ```
   conformability
   3.048000e-01 m
   4.535924e-01 kg
   ```

   The `conformability` message means the units you specified cannot be converted. Feet measure length, and pounds measure mass, so converting from one to the other does not make sense. Therefore, the **units** command displays the equivalent of each value in standard units.

   In other words, this example shows that one foot equals `0.3048` meters and that one pound equals `0.4535924` kilograms. The **units** command shows the equivalents in meters and kilograms because the command considers these units to be the standard measures of length and mass.

Files

/usr/bin/units	Contains the **units** command.
/usr/share/lib/unittab	Lists units that the **units** command creates as well as units defined by the user.

Related Information

The **bc** command, **dc** command.

unlink Command

Purpose

Performs an **unlink** subroutine.

Syntax

unlink *File*

Description

The **unlink** command performs the **unlink** subroutine on a specified file.

The **unlink** command does not issue error messages when the associated subroutine is unsuccessful; you must check the exit value to determine if the command completed normally. It returns a value of 0 if it succeeds, a value of 1 if too few or too many parameters are specified, and a value of 2 if its system call is unsuccessful.

> **Attention:** The **unlink** command allows a user with root user authority to deal with unusual problems, such as moving an entire directory to a different part of the directory tree. It also permits you to create directories that cannot be reached or escaped from. Be careful to preserve the directory structure by observing the following rules:

- Be certain every directory has a . (dot) link to itself.
- Be certain every directory has a .. (dot dot) link to its parent directory.
- Be certain every directory has no more than one link to itself or its parent directory.
- Be certain every directory is accessible from the root of its file system.

An attempt to remove a file or directory that has been exported for use by the NFS version 4 server will fail with a message saying that the resource is busy. The file or directory must be unexported for NFS version 4 use before it can be removed.

Example

To remove a directory entry pointed by `file2`, enter:

```
unlink file2
```

Files

/usr/sbin/unlink	Contains the **unlink** command.

Related Information

The **fsck** command, **link** command, **ln** command.

The **link** subroutine, **unlink** subroutine.

The File systems in *Operating system and device management* explains file system types, management, structure, and maintenance.

The Files in *Operating system and device management* provides information on working with files.

The Directories in *Operating system and device management* explains working with directories and path names.

unloadipsec Command

Purpose

Unloads a crypto module from the IP Security subsystem.

Syntax

unloadipsec -c *crypto_mod_name*

Description

The **unloadipsec** command unloads a crypto module from the IP Security subsystem. The **unloadipsec** command can be used when a crypto module is no longer being used or when a crypto module is to be replaced with a newer version.

A crypto module can only be unloaded after the IP Security device is stopped. The steps for replacing a crypto module are: change the IP Security device to the defined state; unload the old crypto module using this command; uninstall the old module and install the new module, and bring the IP Security device back to the available state.

Flags

-c *crypto_mod_name*	Specifies the name of the crypto module to be unloaded. When used without any flag, the command lists all the crypto modules installed (but not necessarily loaded).

unmirrorvg Command

Purpose

Removes the mirrors that exist on volume groups or specified disks. This command only applies to AIX 4.2.1 or later.

Syntax

unmirrorvg [**-c** *Copies*] *VolumeGroup* [*PhysicalVolume .. .*]

Description

The **unmirrorvg** command unmirrors all the logical volumes detected on a given volume group. This same functionality may also be accomplished manually if you execute the **rmlvcopy** command for each individual logical volume in a volume group.

By default, **unmirrorvg** will pick the set of mirrors to remove from a mirrored volume group. If you wish to control which drives no longer are to contain mirrors, you must include the list of disks in the input parameters, *PhysicalVolume.*

When the *PhysicalVolume* parameter is listed in the command, this indicates that only the mirrors that exist on this disk should be unmirrored. Mirrors that exist on other drives in the volume group, but not listed in a user-provided disk list are left alone and remain mirrored.

Notes:

1. If LVM has not recognized that a disk has failed it is possible that LVM will remove a different mirror. Therefore if you know that a disk has failed and LVM does not show those disks as missing you should specify the failed disks on the command line or you should use **replacepv** to replace the disk or **reducevg** to remove the disk.

2. If a logical volume copy spans more than one disk, the portion of the logical volume copy that resides on a disk not listed by the user is also removed.
3. The **unmirrorvg** command is not allowed on a snapshot volume group.

When **unmirrorvg** is executed, the default COPIES value for each logical volume becomes 1. If you wish to convert your volume group from triply mirrored to doubly mirrored, use the **-c** option.

Note: To use this command, you must either have root user authority or be a member of the **system** group.

Attention: The **unmirrorvg** command may take a significant amount of time to complete because of complex error checking and the number of logical volumes to unmirror in a volume group.

Y ou can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit unmirrorvg
```

Flag

-c *Copies* Specifies the minimum number of copies that each logical volume must have after the **unmirrorvg** command has finished executing. If you do not want all logical volumes to have the same number of copies, then reduce the mirrors manually with the **rmlvcopy** command. If this option is not used, the copies will default to 1.

The following is a description of **rootvg**:

rootvg unmirroring When the **rootvg** unmirroring has completed, you must perform three additional tasks: **bosboot, bootlist**, and **reboot**.

The **bosboot** command is required to reinitialize the boot record on the remaining disk. The **bootlist** command needs to be performed so that the system will only boot to the disk left in **rootvg**. Because unmirroring turns Quorum back on for **rootvg**, a reboot is required for this to take effect.

Examples

1. To unmirror a triply mirrored volume group and leave two copies, enter:

   ```
   unmirrorvg -c 2 workvg
   ```

   The logical partitions in the logical volumes held on workvg now have 2 copies.
2. To get default unmirroring of rootvg, enter:

   ```
   unmirrorvg rootvg
   ```

   rootvg now has only 1 copy.
3. To replace a bad disk drive in a mirrored volume group, enter:

   ```
   unmirrorvg workvg hdisk7
   reducevg workvg hdisk7
   rmdev -l hdisk7 -d
   replace the disk drive, let the drive be renamed hdisk7
   extendvg workvg hdisk7
   mirrorvg workvg
   ```

   Note: By default in this example, **mirrorvg** will try to create 2 copies for logical volumes in workvg. It will try to create the new mirrors onto the replaced disk drive. However, if the original system

had been triply mirrored, there may be no new mirrors created onto hdisk7, as other copies may already exist for the logical volumes. This follows the default behavior of **unmirrorvg** to reduce the mirror copy count to 1.

Note: When **unmirrorvg workvg hdisk7** is run, **hdisk7** will be the remaining drive in the volume group. This drive is not actually removed from the volume group. You must run the **migratepv** command to move the data from the disk that is to be removed from the system to disk **hdisk7**.

Files

/usr/sbin Directory where the **unmirrorvg** command resides.

Related Information

The **migratepv** command, **mklvcopy** command, **mirrorvg** command, **syncvg** command, **reducevg** command, **extendvg** command.

The Logical volume storage in *Operating system and device management*.

---

unpack Command

Purpose

Expands files.

Syntax

unpack *File ...*

Description

The **unpack** command expands files created by the **pack** command. For each file specified, the **unpack** command searches for a file called *File***.z**. If this file is a packed file, the **unpack** command replaces it by its expanded version. The **unpack** command names the new file name by removing the **.z** suffix from *File*. If the user has root authority, the new file has the same access modes, access and modification times, owner, and group as the original file. If the user does not have root authority, the file retains the same access modes, access time, and modification time, but acquires a new owner and group.

The **unpack** command operates only on files ending in **.z**. As a result, when you specify a file name that does not end in **.z**, the **unpack** command adds the suffix and searches the directory for a file name with that suffix.

The exit value is the number of files the **unpack** command was unable to unpack. A file cannot be unpacked if any of the following occurs:

- The file name (exclusive of **.z**) has more than 253 bytes.
- The file cannot be opened.
- The file is not a packed file.
- A file with the unpacked file name already exists.
- The unpacked file cannot be created.

Note: The **unpack** command writes a warning to standard error if the file it is unpacking has links. The new unpacked file has a different i-node than the packed file from which it was created. However, any other files linked to the original i-node of the packed file still exist and are still packed.

Exit Status

This command returns the following exit values:

0 The command ran successfully.
>0 An error occurred.

Example

To unpack packed files:

```
unpack chap1.z chap2
```

This expands the packed files `chap1.z` and `chap2.z`, and replaces them with files named `chap1` and `chap2`. Note that you can give the **unpack** command file names either with or without the **.z** suffix.

Files

/usr/bin/unpack Contains the **unpack** command.

Related Information

The **cat** command, **compress** command, **pack** command.

Files in *Operating system and device management*.

---

untab Command

Purpose

Changes tabs into spaces.

Syntax

untab [*FileName ...*]

Description

The **untab** command reads the file specified by the *FileName* parameter or standard input, and replaces tabs in the input with space characters. If you specify a file with the *FileName* parameter, the **untab** command writes the resulting file back to the original file. If the input is standard input, the **untab** command writes to standard output. The **untab** command assumes that tab stops are set every eight columns, starting with column nine. The file name specified for the *FileName* parameter cannot exceed **PATH_MAX**-9 bytes in length.

Example

To replace tab characters in the `File` file with space characters, enter:

```
untab File
```

Files

/usr/bin/untab Contains the **untab** command.

Related Information

The **expand** command, **newform** command, **tab** command, **unexpand** command.

Files in *Operating system and device management*.

Input and output redirection in *Operating system and device management*.

---

update Command

Purpose

Periodically updates the super block.

Syntax

update

Description

The **update** command executes a **sync** subroutine every 30 seconds. This action ensures the file system is up-to-date in the event of a system crash.

Files

/usr/sbin/update	Contains the **update** command.

Related Information

The **init** command, **rc** command, **sync** command.

The **cron** daemon.

The **sync** subroutine.

---

updatevsdnode Command

Purpose

Modifies virtual shared disk subsystem options.

Syntax

updatevsdnode **–n** {**ALL** | *node_number* [,node_number ...]}
{[**–a** *VSD_adapter*]
[**–b** *min_buddy_buffer_size*]
[**–x** *max_buddy_buffer_size*
[**–s** *max_buddy_buffers*]
[**–M** *vsd_max_ip_packet_size*]}
[**–f**] [**–c** *cluster_name* | NONE]

Description

Use **updatevsdnode** to modify virtual shared disk subsystem options.

Note: This command only modifies the subsystem options. To effectively configure the virtual shared disks, you must first unconfigure all the virtual shared disks, unload the device driver, and then reconfigure the shared disks.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Set/Show Virtual Shared Disk Device Driver Operational Parameters** option or the **Update virtual shared disk Device Driver Node Parameters** option.

Flags

–n Specifies the node numbers of the nodes whose information you want this command to update, or **ALL** nodes in the RSCT peer domain. You can issue the command **/usr/bin/lsclcfg** to find out the node number of the node you are running on.

–a Specifies the adapter name to be used for virtual shared disk communications with this node or nodes. You must specify **ml0** as the adapter name.

–b Specifies the smallest buddy buffer a server uses to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. IBM suggests using the value of 4096 (4KB).

–x The largest buddy buffer a server will use to satisfy a remote request. This value must be a power of 2 and greater than or equal to the *min_buddy_buffer_size*. The recommended value is 262144 (256KB). This value must be the same on all nodes in the RSCT peer domain.

–s This is the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim additional space as needed. The suggested starting value for a 32-bit kernel is 128 256KB buffers. The suggested value is 2000 256KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

Note: The **statvsd** command will indicate if remote requests are queueing waiting for buddy buffers.

–M Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

–f Specifies that this command will force updates to virtual shared disk subsystem options by reconfiguring one or more virtual shared disks on all nodes in the RSCT peer domain on which virtual shared disks are currently configured.

–c *cluster_name* **| NONE**
Changes the cluster the node belongs to. NONE removes the node from the cluster.

Note: The *cluster_name* is required only for SSA (Serial Storage Architecture) disks.

Parameters

vsd_name Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have **root** authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Examples

To increase the buddy buffer size to 48 maximum sized buddy buffers on node 3, enter:

```
updatevsdnode -n 3 -s 48
```

Note: The device driver must be unconfigured from the kernel and reloaded to have this change go into effect.

Location

/opt/lpp/vsd/bin/updatevsdnode

Related Information

Commands: **lsvsd**, **vsdatalst**, **vsdnode**

---

updatevsdtab Command

Purpose

updatevsdtab – Changes the Virtual shared disk subsystem attributes.

Syntax

updatevsdtab {**–v** *vsd_names* | **–a**} {[**–s**]} [**–f**]

Description

Use this command to update the virtual shared disk size. When you change the virtual shared disk size using the **updatevsdtab** command, the change will not take effect until the virtual shared disk is unconfigured and configured again.

If the **–f** flag is specified, the virtual shared disks involved will be reconfigured on all nodes that are up and initially had these virtual shared disks configured.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the Set/Show virtual shared disk Device Driver Operational Parameters option or the Update virtual shared disk Options option.

Flags

–v *vsd_names* Specifies a list of virtual shared disk names to be updated.

–a Specifies that the option is to be changed on all nodes of the system or system partition.

–s Updates the virtual shared disk size after the associated logical volume size is changed.

–f Forces changes by reconfiguring a virtual shared disk on all nodes in the current system partition on which the virtual shared disk is configured.

Parameters

None.

Security

You must have root authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to the *RSCT: Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To reset the size of the virtual shared disk named USER1n3, enter:

```
updatevsdtab -v USER1n3 -s
```

Location

/usr/lpp/csd/bin/updatevsdtab

Related Information

Commands: **defvsd**, **updatevsdnode**

---

updatevsdvg Command

Purpose

Changes virtual shared disk global volume group characteristics.

Syntax

updatevsdvg { **–a** | **–g** *global_volgrp* { **–k VSD –p** *primary_node* **–b** *secondary_node* | **–k CVSD –l** *server_list* [**–c** *cluster_name*] } }

Description

The **updatevsdvg** command changes virtual shared disk global volume group characteristics. This command allows you to change global volume groups from concurrent virtual shared disk volume groups to serial-access (or nonconcurrent) virtual shared disk volume groups, and the other way around. This command can be used whenever server node numbers change, such as replacing or re-cabling servers where the new server numbers are different, or when you need to delete a server.

This command performs the following operations:

1. Suspends all virtual shared disks that are part of this volume group
2. Stops all virtual shared disks that are part of this volume group
3. Issues the **varyoffvg** command for the volume group
4. Verifies that the volume group exists on the new servers and tries to import the volume group if it does not exist
5. Updates the global volume group characteristics
6. Issues the **varyonvg** command for the volume group to the appropriate servers
7. Starts all virtual shared disks that are part of this volume group

Notes:

1. If you issue this command with the **-a** flag, the recoverable virtual shared disk subsystem should not be active. Otherwise, this command can be run while the recoverable virtual shared disk subsystem is active, as long as no application is using the virtual shared disks that are part of the volume group being updated.
2. Concurrent virtual shared disks are supported for disks that have implemented the SCSI-3 persistent reserve model of the AIX SCSI device drivers, and for SSA (Serial Storage Architecture) disks.

Flags

–a
Specifies that persistent reserve information should be reestablished in the object data manager (ODM) for all VSD volume groups served by this node. This flag is intended for the initial setup phase of allowing multiple clusters to access the same virtual shared disks. It is also useful for recovery after the device ODM entries have been removed inadvertently.

This flag causes all of the volume groups served by the node to be varied offline. The volume groups will be varied offline on this node and on all other servers for the volume groups. For this reason, you should stop the recoverable virtual shared disk subsystem before issuing the **updatevsdvg** command with this flag.

–b *secondary_node*
Specifies the secondary node.

–c *cluster_name*
Specifies the cluster name for the server nodes that will be serving concurrently accessed shared disks. This flag is applicable only for SSA (Serial Storage Architecture) disks, and a *cluster_name* must be specified for SSA.

–g *global_volgrp*
Specifies an existing global volume group name.

–k VSD | CVSD
Specifies whether the volume group will be of type concurrent virtual shared disk or serial-access (nonconcurrent) virtual shared disk.

–l *server_list*
Specifies a colon-separated list of servers for concurrent virtual shared disks.

–p *primary_node*
Specifies the primary node.

Parameters

vsd_name Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have **root** authority to run this command.

Exit Status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide*.

Standard Output

Current RVSD subsystem run level.

Examples

1. To change the global volume group named **ess_gvg** from a virtual shared disk global volume group to a concurrent global volume group with three servers, assuming that the disks are cabled correctly and that the disk subsystem supports persistent preserve such as ESS disks, enter:

   ```
   updatevsdvg -g ess_gvg -k CVSD -l 9:17:21
   ```

2. To remove a server from an SSA global volume group named **ssa_gvg**, where the original server list is **9:10** and belongs to an SSA cluster named **cluster9_10**, (that is, the command **vsdatalst -c** shows SSA cluster information), enter:

   ```
   updatevsdvg -g ssa_gvg -k CVSD -l 9 -c cluster9_10
   ```

3. To change a concurrent global volume group named **ess_gvg** back to a virtual shared disk global volume group, where the original server list is **9:17:21**, the new primary node number is 9, and the new secondary node number is 21, enter:

   ```
   updatevsdvg -g ess_gvg -k VSD -p 9 -b 21
   ```

Location

/opt/rsct/vsd/bin/updatevsdvg

Related Information

Commands: **vsdatalst**, **vsdvg**

---

uprintfd Daemon

Purpose

Constructs and writes kernel messages.

Syntax

uprintfd

Description

The **uprintfd** daemon retrieves, converts, formats, and writes kernel messages to processes' controlling terminals. Kernel messages are submitted through the **NLuprintf** and **uprintf** kernel services. Because the **uprintfd** daemon never exits, it should be run only once.

Related Information

The **NLuprintf** kernel service and **uprintf** kernel service.

The Input and Output Handling Programmer's Overview in *AIX 5L Version 5.3 General Programming Concepts: Writing and Debugging Programs*.

---

uptime Command

Purpose

Shows how long the system has been up.

Syntax

uptime

Description

The **uptime** command prints the current time, the length of time the system has been up, the number of users online, and the load average. The load average is the number of runnable processes over the preceding 1-, 5-, 15-minute intervals. The output of the **uptime** command is, essentially, the heading line provided by the **w** command.

Related Information

The **ruptime** command, **w** command.

---

useradd Command

Purpose

Creates a new user account.

Syntax

useradd [**-c** *comment*] [**-d** *dir*] [**-e** *expire*] [**-g** *group*] [**-G** *group1,group2* ...] [**-m** [**-k** *skel_dir*]] [**-u** *uid*] [**-s** *shell*] [**-r** *role1,role2* ...] *login*

Description

The **useradd** command creates a new user account. The *login* parameter must be a unique string (its length is can be configured by administrators using the **chdev** command). You cannot use the `ALL` or `default` keywords in the user name.

The **useradd** command does not create password information for a user. It initializes the **password** field with an asterisk (*). Later, this field is set with the **passwd** or **pwdadm** command. New accounts are disabled until the **passwd** or **pwdadm** commands are used to add authentication information to the **/etc/security/passwd** file.

The **useradd** command always checks the target user registry to make sure the ID for the new account is unique to the target registry. The **useradd** command can also be configured to check all user registries of

the system using the **dist_uniqid** system attribute. The **dist_uniqid** system attribute is an attribute of the **usw** stanza of the **/etc/security/login.cfg** file, and can be managed using the **chsec** command.

The **dist_uniqid** system attribute has the following values:

never Does not check for ID collision against the nontarget registries. This is the default setting.

always
Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry, account creation or modification fails.

uniqbyname
Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the **dist_uniqid** system attribute.

The **uniqbyname** system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the **useradd** command is unspecified when creating a new account in a third registry using colliding ID values. The new account creation might succeed or fail depending on the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries, or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The **useradd** command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

Flags

-c *comment*	Supplies general information about the user specified by the *login* parameter. The *comment* parameter is a string with no embedded colon (:) characters and cannot end with the characters '#!'.
-d *dir*	Identifies the home directory of the user specified by the *login* parameter. The *dir* parameter is a full path name.
-e *expire*	Identifies the expiration date of the account. The *expire* parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* is the month, *DD* is the day, *hh* is the hour, *mm* is the minute, and *yy* is the last 2 digits of the years 1939 through 2038. All characters are numeric. If the *expire* parameter is 0, the account does not expire. The default is 0. See the **date** command for more information.
-g *group*	Identifies the user's primary group. The *group* parameter must contain a valid group name and cannot be a null value.
-G *group1,group2,...*	Identifies the groups the user belongs to. The *group1,group2,...* parameter is a comma-separated list of group names.
-k *skel_dir*	Copies default files from *skel_dir* to user's home directory. Used only with **-m** flag.
-m	Makes user's home directory if it does not exist. The default is not to make the home directory.
-r *role1,role2,...*	Lists the administrative roles for this user. The *role1,role2,...* parameter is a list of role names, separated by commas.

-s *shell* Defines the program run for the user at session initiation. The *shell* parameter is a full path name.

-u *uid* Specifies the user ID. The *uid* parameter is a unique integer string. Avoid changing this attribute so that system security will not be compromised.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To create the `davis` user account with default values, enter:

   ```
   useradd davis
   ```

Restrictions

To prevent login inconsistencies, avoid composing user names entirely of uppercase alphabetic characters. While the **useradd** command supports multibyte user names, restrict user names to characters with the POSIX-portable filename character set.

To ensure that your user database remains uncorrupted, you must be careful when naming users. User names must not begin with a hyphen (-), plus sign (+), at sign (@), or tilde (~). You cannot use the keywords `ALL` or `default` in a user name. Additionally, do not use any of the following characters within a user-name string:

: Colon

" Double quote

Pound sign

, Comma

= Equal sign

**** Back slash

/ Slash

? Question mark

' Single quote

` Back quote

Finally, the *login* parameter cannot contain any space, tab, or newline characters.

Location

/usr/sbin/useradd

Files

The **useradd** command has read and write permissions to the following files.

/etc/passwd Contains the basic attributes of users.

/etc/security/user Contains the extended attributes of users.

/etc/security/user.roles Contains the administrative role attributes of users.

/etc/security/limits Defines resource quotas and limits for each user.

/etc/security/environ Contains the environment attributes of users.

/etc/security/audit/config Contains audit configuration information.

/etc/security/lastlog Contains the last login attributes of users.

/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

Related Information

The chfn Command, chgroup Command, chgrpmem Command, chsh Command, lsgroup Command, lsuser Command, mkgroup Command, mkuser Command, passwd Command, pwdadm Command, rmgroup Command, rmuser Command, “setgroups Command” on page 74, “setsenv Command” on page 79, “su Command” on page 238, “userdel Command,” “usermod Command” on page 588.

---

userdel Command

Purpose

Removes a user account.

Syntax

userdel [**-r**] *login*

Description

The **userdel** command removes the user account identified by the *login* parameter. The command removes a user’s attributes without removing the user’s home directory by default. The user name must already exist. If the **-r** flag is specified, the **userdel** command also removes the user’s home directory.

Only the root user or users with **UserAdmin** authorization can remove administrative users. Administrative users are those users with `admin=true` set in the **/etc/security/user** file.

Flags

-r	Removes the user's home directory.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Examples

1. To remove the user `davis` account and its attributes from the local system, enter:

   ```
   userdel davis
   ```

Location

/usr/sbin/userdel

Files

The **userdel** command has read and write permissions to the following files.

/etc/passwd	Contains the basic attributes of users.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/environ	Contains the environment attributes of users.

/etc/security/audit/config	Contains audit configuration information.
/etc/security/lastlog	Contains the last login attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

Related Information

The chfn Command, chgroup Command, chgrpmem Command, chsh Command, lsgroup Command, lsuser Command, mkgroup Command, mkuser Command, passwd Command, pwdadm Command, rmgroup Command, rmuser Command, “setgroups Command” on page 74, “setsenv Command” on page 79, “su Command” on page 238, “useradd Command” on page 584, “usermod Command.”

---

usermod Command

Purpose

Changes user attributes.

Syntax

usermod [**-u** *uid*] [**-G** *group1,group2 ...*] [**-d** *dir* [**-m**]] [**-s** *shell*] [**-c** *comment*] [**-l** *new_name*] [**-e** *expire*] [**-R** *role1,role2 ...*] *login*

Description

Attention: Do not use the **usermod** command if you have a Network Information Service (NIS) database installed on your system.

The **usermod** command changes attributes for the user identified by the *login* parameter. The user name must already exist. To change an attribute, specify the flag and the new value. The following files contain local user attributes that are set by this command:

- **/etc/passwd**
- **/etc/security/environ**
- **/etc/security/limits**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/audit/config**
- **/etc/group**
- **/etc/security/group**

Avoid changing the ID for an account so that system security is not compromised. However, when the ID is changed using the **usermod** command, ID collision checking is also controlled by the **dist_uniqid** attribute in the **usw** stanza of the **/etc/security/login.cfg** file. The behavior of ID collision control is the same as that described for the **mkuser** command.

Flags

-c *comment*	Supplies general information about the user specified by the *login* parameter. The *comment* parameter is a string with no embedded colon (:) characters and cannot end with the characters `'#!'`.
-d *dir*	Changes the home directory to the directory specified by the *dir* parameter.

-e *expire*	Identifies the expiration date of the account. The *expire* parameter is a 10-character string in the *MMDDhhmmyy* form, where *MM* is the month, *DD* is the day, *hh* is the hour, *mm* is the minute, and *yy* is the last 2 digits of the years 1939 through 2038. All characters are numeric. If the *expire* parameter is 0, the account does not expire. The default is 0. See the **date** command for more information.
-G *group1,group2,...*	Identifies the groups the user belongs to. The *group1,group2,...* parameter is a comma-separated list of group names.
-l *new_name*	Specifies the new name of the user.
-m	Moves the contents of the user's current home directory to the new home directory. Only used with the **-d** flag.
-R *role1,role2,...*	Lists the administrative roles for this user. The *role1,role2,...* parameter is a list of role names, separated by commas.
-s *shell*	Defines the program run for the user at session initiation. The *shell* parameter is a full path name.
-u *uid*	Specifies the user ID. The *uid* parameter is a unique integer string. Avoid changing this attribute so that system security will not be compromised.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Examples

1. To change the user `davis` to be a member of the `system` group, enter:

   ```
   usermod -g system davis
   ```

Restrictions

To ensure the integrity of user information, some restrictions apply when using the usermod command. Only the root user or users with **UserAdmin** authorization can use the **usermod** command to perform the following tasks:

- Make a user an administrative user by setting the admin attribute to true.
- Change any attributes of an administrative user.
- Add a user to an administrative group

An administrative group is a group with the **admin** attribute set to True. Members of the security group can change the attributes of nonadministrative users and add users to nonadministrative groups.

The **usermod** command manipulates local user data only. You cannot use it to change data in registry servers like NIS and DCE.

Location

/usr/sbin/usermod

Files

The **usermod** command has read and write permissions to the following files.

/etc/passwd	Contains the basic attributes of users.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/audit/config	Contains audit configuration information.
/etc/security/lastlog	Contains the last login attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.

Related Information

The chfn Command, chgroup Command, chgrpmem Command, chsh Command, lsgroup Command, lsuser Command, mkgroup Command, mkuser Command, passwd Command, pwdadm Command, rmgroup Command, rmuser Command, “setgroups Command” on page 74, “setsenv Command” on page 79, “su Command” on page 238, “useradd Command” on page 584, “userdel Command” on page 587.

users Command

Purpose

Displays a compact list of the users currently on the system.

Syntax

users [*File*]

Description

The **users** command lists the login names of the users currently on the system to standard output (**stdout**) in a compact, one-line list format. If an argument is given, it is used as an alternate file instead of **/etc/utmp**.

Files

/etc/utmp	Contains list of current users.
/usr/bin/users	Contains the **users** command.

Related Information

The **who** command.

usrck Command

Purpose

Verifies the correctness of a user definition.

Syntax

usrck { **-l** [**-b**] | **-n** | **-p** | **-t** | **-y** } { **ALL** | *User ...* }

Description

The **usrck** command verifies the correctness of the user definitions in the user database files, by checking the definitions for **ALL** the users or for the users specified by the *User* parameter. If more than one user is specified, there must be a space between the names. You must select a flag to indicate whether the system should try to fix erroneous attributes.

The command first checks the entries in the **/etc/passwd** file. If you indicate that the system should fix errors, duplicate user names are reported and removed. Duplicate IDs are reported only, because there is no system fix. If an entry has fewer than six colon-separated fields, the entry is reported, but not fixed. The **usrck** command next checks specific user attributes in other files.

The **usrck** command verifies that each user name listed in the **/etc/passwd** file has a stanza in the **/etc/security/user**, **/etc/security/limits** and **/etc/security/passwd** files. The **usrck** command also verifies that each group name listed in the **/etc/group** file has a stanza in the **/etc/security/group** file. The **usrck** command using the **-y** flag creates stanzas in the security files for the missing user and group names.

Note: This command writes its messages to **stderr**.

A list of all the user attributes follows, with notations stating which attributes are checked:

account_locked	No check. The **usrck** command sets this attribute to True and disables accounts.
admgroups	Checks to see if the **admgroups** are defined in the user database and, if you indicate that the system should fix errors, the command removes any groups that are not in the database.
auditclasses	Checks to see if the **auditclasses** are defined for the user in the **/etc/security/audit/config** file. If you indicate that the system should fix errors, the command deletes all the auditclasses that are not defined in the **/etc/security/audit/config** file.
auth1	Checks the primary authentication method. Unless the method is NONE or SYSTEM, it must be defined in the **/etc/security/login.cfg** file and the program attribute must exist and be executable by the root user. If you indicate that the system should fix errors, it will disable the user account if an error is found.
auth2	Checks the secondary authentication method. Unless the method is NONE or SYSTEM, it must be defined in the **/etc/security/login.cfg** file and the program attribute must exist and be executable by the root user. There is no system fix.
core	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
core_hard	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
cpu	Ensures that the values are sensible. If not, the command resets the values to 120 seconds, the minimum value.
cpu_hard	Ensures that the values are sensible. If not, the command resets the values to 120 seconds, the minimum value.
data	Ensures that the values are sensible. If not, the command resets the values to 1272 blocks (636K), the minimum value.
data_hard	Ensures that the values are sensible. If not, the command resets the values to 1272 blocks (636K), the minimum value.
dictionlist	Checks the list of dictionary files. If you indicate that the system should fix errors, all dictionary files that do not exist are deleted from the user database.
expires	No check.
fsize	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
fsize_hard	Ensures that the values are sensible. If not, the command resets the values to 200 blocks, the minimum value.
gecos	No check.
histexpire	Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

histsize Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

home Checks the existence and accessibility of the home directory by read mode and search mode. If you indicate that the system should fix errors, it will disable the user account if an error is found.

id Checks the uniqueness of the user ID. If you indicate that the system should fix errors, the command deletes any invalid entry in the **/etc/passwd** file.

login No check.

loginretries Checks if the user attempted unsuccessful logins more than the allowable amount. If so, the system disables the user account.

logintimes Ensures that the string of time specifiers is valid. If you indicate that the system should fix errors, the system disables the user account if an error is found.

maxage Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

maxexpired Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

maxrepeats Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

minage Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value. The system also indicates if the **minage** attribute is larger than the **maxage** attribute.

minalpha Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

mindiff Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

minlen Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value.

minother Ensures that the values are sensible. If you indicate that the system should fix errors, values that are too large are set to the largest possible value and values that are too small are set to the smallest possible value. The system also indicates if the **minage** attribute plus the **maxage** attribute is greater than the maximum password size.

name Checks the uniqueness and composition of the user name. The name must be a unique string of eight bytes or less. It cannot begin with a + (plus sign), a : (colon), a - (minus sign), or a ~ (tilde). Names beginning with a + (plus sign) or with a - (minus sign) are assumed to be names in the NIS (Network Information Service) domain, and no further processing is performed. It cannot contain a colon (:) in the string and cannot be the **ALL** or **default** keywords. If you indicate that the system should fix errors, the command disables the user account if an error is found and deletes any invalid entry in the **/etc/passwd** file.

The **usrck** command verifies that, for each user name listed in the **/etc/passwd** file, there is a stanza in the **/etc/security/user**, **/etc/security/limits**, and **/etc/security/passwd** files. The command adds stanzas for each one identified as missing. The **usrck** command additionally verifies that each group name listed in the **/etc/group** file has a stanza in the **/etc/security/group** file.

nofiles Ensures that the value is sensible. If not, resets the value to 200, the minimum value.

nofiles_hard Ensures that the value is sensible. If not, resets the value to 200, the minimum value.

pgrp Checks for the existence of the primary group in the user database. If you indicate that the system should fix errors, it will disable the user account if an error is found.

pwdchecks	Checks the list of external password restriction methods. If you indicate that the system should fix errors, all methods that do not exist are deleted from the user database.
pwdwarntime	Ensures that the value is sensible. If not, the system resets the value to the difference between the **maxage** and **minage** values.
rlogin	No check.
rss	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64K), the minimum value.
rss_hard	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64K), the minimum value.
shell	Checks the existence and accessibility of the shell by execute mode. If you indicate that the system should fix errors, it will disable the user account if an error is found.
stack	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64K), the minimum value.
stack_hard	Checks to ensure that the values are sensible. If not, the command resets the values to 128 blocks (64K), the minimum value.
su	No check.
sugroups	Checks for the existence of the **sugroups** in the user database files. If you indicate that the system should fix errors, it will delete all the groups that are not in the database.
sysenv	No check.
tpath	Checks to ensure that the **shell** attribute is tagged as a trusted process if **tpath=always**. If you indicate that the system should fix errors, it will disable the user account if an error is found.
ttys	Checks for the existence of the ttys in the user database files. If you indicate that the system should fix errors, it will delete all the ttys that do not exist from the user database.
usrenv	No check.

If the fix involves disabling a user account, use the **chuser** command to reset the value of the **account_locked** attribute to False. You can use the System Management Interface Tool (SMIT) to run the **chuser** command by entering:

```
smit chuser
```

The root user or a member of the security group can enable a user account again by removing the **account_locked** attribute or setting the **account_locked** attribute to False. The root user's account is not disabled by the **usrck** command.

Generally, the **sysck** command calls the **usrck** command as part of the verification of a trusted-system installation. If the **usrck** command finds any errors in the user database, the root user or a member of the security group should execute both the **grpck** command and the **pwdck** command.

The **usrck** command checks to see if the database management security files (**/etc/passwd.nm.idx**, **/etc/passwd.id.idx**, **/etc/security/passwd.idx**, and **/etc/security/lastlog.idx**) files are up-to-date or newer than the corresponding system security files. Please note, it is all right for the **/etc/security/lastlog.idx** to be not newer than **/etc/security/lastlog**. If the database management security files are out-of-date, a warning message appears indicating that the root user should run the **mkpasswd** command.

The **usrck** command checks if the specified user can log in. If the user cannot log in because of too many unsuccessful login attempts or because the password is expired, the **usrck** command issues a warning message indicating why the user cannot log in. If you indicate that the system should fix errors, the system disables the user account if the user cannot log in for the above reasons.

If the **-l** flag is specified, the **usrck** command scans all users or the users specified by the *User* parameter to determine if users can access the system. The criteria used to determine accessibility for a user are listed in the following table:

Table 2. User Accessibility Criteria

Criterion	Description	Cause
1	User account is locked.	The user's **account_locked** attribute is set to **true**.
2	User account is expired.	The user's **expires** attribute is set to a value (expiration time) that is expired.
3	User has too many consecutive failed login attempts.	The user's **unsuccessful_login_count** value is greater than the user's **loginretries** value.
4	User has no password.	The user's **password** field is '*' in **/etc/password** or **/etc/security/password**.
5	User is not allowed to log in for this date/time.	The current date/time is not within the allowed time as defined by the user's **logintimes** attribute.
6	The **/etc/nologin** file exists.	The **/etc/nologin** file prevents a non-root user from logging in.
7	User password is expired and only system administrator can change it.	The user's password is expired and the ADMIN password flag is set.
8	User is denied login to host.	The user's **hostallowedlogin** and **hostsdeniedlogin** attributes do not allow access to the current host.
9	User is denied access by applications.	The user's **login**, **rlogin**, and **su** attributes are set to false and the **rcmds** attribute is set to deny. If at least one but not all of these attribute values deny authorization, the system is considered partially accessible by the user.
10	User is denied login to terminal.	The user's **ttys** attribute does not allow access to the current terminal. The system is considered partially accessible for the user.

If the **-b** flag is also specified, the output consists of two fields, the user name and a 16-digit bit mask, separated by a tab. Each digit in the bit mask corresponds to a criteria in the User Accessibility Criteria table above, with criteria 1 represented by the rightmost digit. If the bit location for a criteria is set to 1, the check for this criteria failed for the user. Extra digits in the output are reserved for future use.

The following is an example of the **usrck** command with the **-l** flag:

```
# usrck -l testusr1 testusr2
3001-689 The system is inaccessible to testusr1, due to the following:
         User account is locked
         User denied login to terminal.

3001-689 The system is inaccessible to testusr2, due to the following:
         User account is expired.
         User has too many consecutive failed login attempts.
         User denied login to host.
```

The following is an example of the **usrck** command with the **-l** and **-b** flags:

```
# usrck -lb testusr1 testusr2
 testusr1       0000000000000001
 testusr2       0000000001000110
```

Flags

-b Reports users who are not able to access the system and the reasons, with the reasons displayed in a bit-mask format. The **-l** flag must be specified if the **-b** flag is specified.
Note: The bit mask does not report criteria 10 (user denied login to terminal), since this cannot be considered a complete scenario when determining if a system is inaccessible to a user. Likewise, the bit mask does not report criteria 9 (User denied access by applications) if at least one but not all of the attributes' values deny authentication; this criteria is only reported when all four attribute values deny authentication.

-l Scans all users or the users specified by the *User* parameter to determine if the users can access the system.

-n Reports errors but does not fix them.

-p Fixes errors but does not report them.

-t Reports errors and asks if they should be fixed.

-y Fixes errors and reports them.

Security

Access Control: This command should grant execute (x) access to the root user and members of the security group. The command should be **setuid** to the root user and have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/security/user**
rw	**/etc/security/group**
rw	**/etc/group**
rw	**/etc/security/lastlog**
rw	**/etc/security/limits**
rw	**/etc/security/audit/config**
rw	**/etc/security/login.cfg**

Auditing Events:

Event	Information
USER_Check	user, attribute-error, status

Exit Status

This command returns the following exit values:

0 User definition files are appropriate.

>0 An error occurred or there is an error in one or more user definition files. The following error codes are returned:

EINVAL (22)
Invalid command line arguments

ENOENT (2)
One or more user definition files do not exist

ENOTRUST (114)
Errors in user definitions in the database files or users unable to access the system (found by **-l** option)

Examples

1. To verify that all the users exist in the user database, and have any errors reported (but not fixed), enter:

   ```
   usrck  -n ALL
   ```

2. To delete from the user definitions those users who are not in the user database files, and have any errors reported, enter:

   ```
   usrck  -y ALL
   ```

3. To display the list of users who are unable to access the system, enter:

   ```
   usrck -l ALL
   ```

4. To display the list of users who are unable to access the system, in a bit mask format, enter:

   ```
   usrck -l -b ALL
   ```

Files

/usr/bin/usrck	Specifies the path of the **usrck** command.
/etc/passwd	Contains basic user attributes.
/etc/security/user	Contains the extended attributes of users.
/etc/group	Contains basic group attributes.
/etc/security/group	Contains the extended attributes of groups.
/etc/security/lastlog	Contains the last login attributes for users.
/etc/security/limits	Contains the process resource limits of users.
/etc/security/audit/config	Contains audit system configuration information.
/etc/security/login.cfg	Contains configuration information.

Related Information

The **grpck** command, **pwdck** command, **sysck** command.

Securing the network in *Security* describes the identification and authentication of users, discretionary access control, the trusted computing base, and auditing.

---

utmpd Daemon

Purpose

Monitors and maintains **/etc/utmp** file.

Syntax

/usr/sbin/utmpd [*Interval*]

Description

The **utmpd** daemon monitors the **/etc/utmp** file for validity of the user process entries at regular intervals. An user process that has been terminated, but has not been cleaned up in the **/etc/utmp** file, is removed by cross checking the process id of the entry against the process table.

The Interval parameter specifies the amount of time in seconds between each scan of the **/etc/utmp** file. The default interval time would be 300 seconds.

Usage

To start **utmpd** from **/etc/inittab**, add the following entry to the file:

```
utmpd:2:respawn:/usr/sbin/utmpd
```

init starts the **utmpd** daemon during system startup. To have the changes take effects immediately without rebooting, type:

```
telinit q
```

Security

Only the root user can read and execute this command.

Files

/etc/inittab	Specifies stanzas read by the **init** command.
/etc/utmp	Contains a record of users logged into the system.

---

uucheck Command

Purpose

Checks for files and directories required by BNU.

Syntax

uucheck [**-v**] [**-x** *DebugLevel*]

Description

The **uucheck** command verifies the presence of the files and directories required by the Basic Networking Utilities (BNU) facility. The command also checks for some errors in the **/etc/uucp/Permissions** file.

> **Note:** The **uucheck** command does not check for correct file and directory modes or for errors in the **/etc/uucp/Permissions** file, such as duplicate login or machine names.

Issue the **uucheck** command from the command line after installing the BNU program, configuring the BNU facility for your site, or making changes in part of the BNU facility, such as the **/etc/uucp/Permissions** file.

> **Note:** Only someone with root user authority can use the **uucheck** command at the command line.

Flags

-v	Displays a detailed explanation of how BNU interprets the **/etc/uucp/Permissions** file.

-x*DebugLevel* Displays debugging information. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. The higher the number, the more detailed the information.

Examples

1. To find out how the BNU programs interpret the **/etc/uucp/Permissions** file, enter:

   ```
   uucheck  -v
   ```

   The **-v** flag instructs the **uucheck** command to verify that the BNU files exist and displays a detailed explanation of how the BNU programs interpret the **/etc/uucp/Permissions** file. The output is similar to the following:

   ```
   *** uucheck: Check Required Files and Directories
   *** uucheck: Directories Check Complete

   *** uucheck:  Check /etc/uucp/Permissions file
   ** LOGNAME PHASE (when they call us)

   When a system logs in as: (unostro)
      We DO allow them to request files.
      We WILL send files queued for them on this call.
      They can send files to
         /
      They can request files from
         /
      Myname for the conversation will be plague.austin..
      PUBDIR for the conversation will be
      /var/spool/uucppublic.

   ** MACHINE PHASE (when we call or execute their uux requests)

   When we call system(s): (nostromo)
      We DO allow them to request files.
      They can send files to
         /
      They can request files from
         /
      Myname for the conversation will be plague.austin..
      PUBDIR for the conversation will be
      /var/spool/uucppublic.

   Machine(s): (nostromo)
   CAN execute the following commands:
   command (ALL), fullname (ALL)

   *** uucheck:  /etc/uucp/Permissions Check Complete
   ```

   For an explanation of these permissions, see the **/etc/uucp/Permissions** file.

2. To debug with the **uucheck** command, enter:

   ```
   uucheck  -x8
   ```

   The -x8 flag produces extensive debugging output.

Files

/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.

Related Information

The **uucp** command, **uustat** command, **uux** command.

The **uucico** daemon, **uusched** daemon.

How to Configure BNU in *Networks and communication management*.

---

uucico Daemon

Purpose

Transfers Basic Networking Utilities (BNU) command, data, and execute files to remote systems.

Syntax

uucico [**-r** *RoleNumber*] [**-x** *DebugLevel*] **-s** *SystemName*

Description

The **uucico** daemon transfers Basic Networking Utilities (BNU) command (**C.***), data (**D.***), and execute (**E.***) files, created by the **uucp** and **uux** commands, to a specified remote system. Both the local and remote systems run the **uucico** daemon, and the two daemons communicate with each other to complete transfer requests.

The **uucico** daemon performs the following actions:

1. Scans the spooling directory (**/var/spool/uucp/***SystemName*) on the local system for transfer requests.
2. Selects the device used for the communications connection after checking the **/etc/uucp/Devices** file and the lock files in the **/etc/locks** directory.
3. Places a call to the specified remote system using information in the **Systems**, **Dialers**, and **Dialcodes** files located in the **/etc/uucp** directory.
4. Performs the required login sequence specified in the **Systems** file.
5. Checks permissions listed in the **/etc/uucp/Permissions** file.
6. Checks scheduling limits in the **Maxuuscheds** and **Maxuuxqts** files located in the **/etc/uucp** directory.
7. Runs all transfer requests from both the local and the remote system, placing the transferred files in the public directories (**/var/spool/uucppublic/***).
8. Logs transfer requests and completions in files in the **/var/spool/uucp/.Log/uucico** directory.
9. Notifies specified users of transfer requests.

Usually the **uucico** daemon is called by the **uucp** and **uux** commands when needed and is started periodically by the BNU scheduling daemon, **uusched**, which is started by the **cron** daemon.

The **uucico** daemon can be started from the command line for debugging. The BNU **uutry**, **Uutry**, and **uukick** commands also start the **uucico** daemon with debugging turned on.

Notes:

1. Either you must be in the **/usr/sbin/uucp** directory when you call the **uucico** daemon, or you must call the daemon with the full path name, **/usr/sbin/uucp/uucico**.
2. In the case of a **uux** command request for the execution of a command on a remote system, the **uucico** daemon transfers the files and the **uuxqt** daemon executes the command on the remote system.

Flags

-r *RoleNumber*	Specifies the server and client relationship. The role numbers are 1 for server mode and 0 for client mode. If the **-r** flag is not used, the **uucico** daemon is started in client mode (**-r** 0), because the **uucico** daemon is generally started automatically by a BNU command or daemon. When the **uucico** daemon is started manually, this flag should be set to 1.
-x *DebugLevel*	Displays debugging information on the screen of the local terminal. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the information to be more detailed. This flag is useful for diagnosing problems with the expect-send sequence in the **/etc/uucp/Systems** file.
-s *SystemName*	Specifies the name of the remote system. This flag is required when starting the **uucico** daemon from the command line. The *SystemName* variable is supplied internally when the **uucico** daemon is started automatically.

Note: System names must contain only ASCII characters.

Example

To call the **uucico** daemon from the command line, enter:

```
/usr/sbin/uucp/uucico  -r 1  -s hera &
```

to start the daemon as a background process and contact remote system `hera`.

Files

/etc/locks /*	Contains lock files which prevent multiple uses of devices and multiple calls to systems.
/usr/sbin/uucp/*	Contains the **uucico** daemon and the configuration files for BNU.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Maxuuscheds	Limits scheduled jobs.
/etc/uucp/Maxuuxqts	Limits remote command executions.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/.Admin/errors	Lists **uucico** daemon errors that BNU cannot correct.
/var/spool/uucp/.Log/uucico /*	Contains **uucico** daemon log files.
/var/spool/uucp/.Status/*SystemName*	Lists the last time a remote system was contacted and the minimum time until the next retry.
/var/spool/uucp/*SystemName* **/***	Contains **C.***, **D.***, and **X.*** files to be transferred by the **uucico** daemon.
/var/spool/uucp/*SystemName***/C.***	Contains command files.
/var/spool/uucp/*SystemName***/D.***	Contains data files.
/var/spool/uucp/*SystemName***/X.***	Contains execute files.
/var/spool/uucppublic/*	Contain files after transfer by the **uucico** daemon.

Related Information

The **cron** daemon, **uusched** daemon.

The **uucp** command, **uukick** command, **uuq** command, **uustat** command, **uusnap** command, **uutry** command, **Uutry** command, **uux** command.

Monitoring a BNU remote connection, Monitoring a BNU file transfer, Debugging BNU login failures using the uucico daemon in *Networks and communication management*.

BNU daemons, BNU file and directory structure in *Networks and communication management*.

---

uuclean Command

Purpose

Removes files from the BNU spool directory.

Syntax

/usr/sbin/uucp/uuclean [**-m**] [**-n***Hours*] [**-p***Prefix*] [**-d***Subdirectory*]

Description

The **uuclean** command checks the Basic Networking Utilities (BNU) spool directory (**/var/spool/uucp**) for files with the specified prefixes and deletes those that are older than the given number of hours. If the **-n***Hours* flag is not included, the **uuclean** command deletes files that are older than 72 hours.

If the **-p** flag is not included, the **uuclean** command deletes all files in the specified subdirectories of the spool directory that meet the age requirement. If the **-d** flag is not included, the command deletes all the files (that meet the age and prefix requirements) in all the subdirectories of the spool directory. Thus if neither the **-d** or the **-p** flag is included, the **uuclean** command deletes *all* files in *all* subdirectories of the **/var/spool/uucp** directory that meet the age requirement.

If the **-m** flag is not specified, the **uuclean** command sends mail to owners of all command (**C.***) files that it deletes. If the **-m** flag is specified, the command sends mail to the owner of each file it deletes, including data (**D.***) and execute (**X.***) files. The mail message includes the name of the deleted file.

The **uuclean** command is usually run by the **cron** daemon.

> **Note:** Only someone with root user authority or who is logged in as **uucp** can issue the **uuclean** command.

Flags

-d*Subdirectory*	Deletes files from the specified subdirectory of the **/var/spool/uucp** directory if they match specifications given with the **-n** and **-p** flags. If the **-d** flag is not specified, the **uuclean** command checks all subdirectories of the **/var/spool/uucp** directory. Up to 10 subdirectories can be specified with the **-d** flag.
-m	Instructs the **uuclean** command to send mail to the owner of each file when it is deleted.
-n*Hours*	Deletes files whose ages are more than the number of hours specified by the *Hours* variable, if they match specifications given with the **-d** and **-p** flags. The default is 72 hours.
-p*Prefix*	Deletes files with the prefix given by the *Prefix* variable, if they match specifications given with the **-n** and **-d** flags. Up to 10 prefixes can be specified with the **-p** flag.

Examples

1. To delete all old command files, enter:

   ```
   /usr/sbin/uucp/uuclean -pC
   ```

   This command deletes all files in all subdirectories of the **/var/spool/uucp** directory whose names begin with `C` and that are older than 72 hours (the default). The system sends mail to the original owner of each file, stating that the file has been deleted.

2. To delete all old files from the spool directory for systems `venus` and `nostromo`, enter:
   ```
   /usr/sbin/uucp/uuclean -n84 -dvenus -dnostromo
   ```

   This command deletes all files in the **/var/spool/uucp/venus** and **/var/spool/uucp/nostromo** directories that are older than 84 hours. By default, the system notifies owners of **C.*** files that the files have been deleted; however, it does not notify owners of other files it deletes.
3. To delete all old files from all spool directories and notify users that they have been deleted, enter:
   ```
   /usr/sbin/uucp/uuclean -m
   ```

   This command deletes all files in all subdirectories of the spool directory, if the files are older than 72 hours (the default). It sends mail to the owner of each file it deletes.
4. To schedule the **uuclean** command to be started periodically by the **cron** daemon, add an entry similar to the following to your **/var/spool/cron/crontabs/uucp** file:
   ```
   15 22 * * * /usr/sbin/uucp/uuclean -n96 -pC -pD -pX
   ```

   This entry will cause the **cron** daemon to start the **uuclean** command at 22:15 (10:15 p.m.) daily. The **uuclean** command will delete all command (**C.***), data (**D.***), and execute (**X.***) files that are older than 96 hours from all subdirectories of the spool directory.

Files

/usr/sbin/uucp/uuclean	Contains the **uuclean** command.
/var/spool/uucp /*	Contains spooling files removed by the **uuclean** command.
/var/spool/cron/crontabs/uucp	Schedules **uucp** jobs for the **cron** daemon.

Related Information

The **uucp** command, **uux** command, **uucleanup** command, **uudemon.cleanu** command.

The **uucico** daemon, **cron** daemon.

BNU file and directory structure, BNU maintenance commands in *Networks and communication management*.

---

uucleanup Command

Purpose

Deletes selected files from the Basic Networking Utilities (BNU) spooling directory.

Syntax

uucleanup [**-C***Days*] [**-W***Days*] [**-m***String*] [**-D***Days*] [**-T***Days*] [**-X***Days*] [**-o** *Days*] [**-s***SystemName*]

Description

The Basic Networking Utilities (BNU) **uucleanup** command scans the spooling directory (**/var/spool/uucp**) for files that are older than a specified number of days and removes them. The **uucleanup** command performs the following tasks:

- Informs the requester of send and receive requests for systems that cannot be reached.
- Warns users about requests that have been waiting for a given number of days. The default is 1 day.
- Returns to the sender mail that cannot be delivered.
- Removes from the spool directory all other files older than a specified number of days.

Notes:

1. The **uucleanup** command is not usually entered on the command line but is executed by the **uudemon.cleanu** command, a shell procedure. Only someone with root user privileges can issue the **uucleanup** command from the command line.
2. When BNU is installed, automatic cleanup is not enabled. Edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (#) from the beginning of the **uudemon.cleanu** line to instruct the **cron** daemon to start the **uudemon.cleanu** command.

Flags

-C*Days*	Removes **C.*** (command) files as old as, or older than, the number of days specified by the *Days* variable, and notifies the requester that the files have been deleted. The default time is 7 days.
-D*Days*	Removes **D.*** (data) files as old as, or older than, the number of days specified by the *Days* variable. Also attempts to deliver any remaining mail messages. The default time is 7 days.
-m*String*	Includes a specified line of text in the warning message generated by the **-W***Days* option. The default line is `See your local administrator to locate the problem.`
-o*Days*	Removes other files as old as, or older than, the number of days specified by the *Days* variable. The default time is 2 days.
-s*SystemName*	Executes the **uucleanup** command only on the spooling directory specified by the *System* variable. The default is to clean up all BNU spooling directories. **Note:** System names can contain only ASCII characters.
-T*Days*	Removes **TM.*** (temporary) files as old as, or older than, the number of days specified by the *Days* variable. Also attempts to deliver any remaining mail messages. The default time is 7 days.
-W*Days*	Sends an electronic mail message to the requester warning that **C.*** (command) files as old as, or older than, the number of days specified by the *Days* variable are still in the spooling directory. The message includes the job ID and, if the request included mail, the mail message. The administrator can use the **-m** option to include a message line telling whom to call to check the problem. The default time is 1 day.
-X*Days*	Removes any **X.*** (execute) files as old as, or older than, the number of days specified by the *Days* variable. The default time is 2 days.

Examples

Warning Users That Their Command Files Have Not Been Sent

1. To send a warning for **C.*** (command) files 2 or more days old, enter:

   ```
   uucleanup  -W2
   ```

   This warns the requester that the files have not been sent.
2. To send a message with the warning, enter:

   ```
   uucleanup  -m"Check these files waiting in the BNU job queue."
   ```

   This locates **C.*** (command) files 1 or more days old (default), warns requesters that their files have not been sent, and gives the message: `Check these files waiting in the BNU job queue.`

Cleaning Up Command, Data, Execute, and Miscellaneous Files

1. To clean up command files 5 or more days old, enter:

   ```
   uucleanup  -C5
   ```

This removes all **C.*** (command) files 5 or more days old and sends an appropriate message to the requesters.

2. To clean up data and execute files 3 or more days old, enter:

```
uucleanup  -D3  -X3
```

   This removes all **D.*** (data) files and all **X.*** (execute) files 3 or more days old.

3. To clean up all files at once using defaults, enter:

```
uucleanup
```

   This removes all **C.***, **D.***, **T.***, and **X.*** files, and all other files older than the default times.

> **Note:** Whenever the **-C** and **-W** flags are used together, make sure the value specified for the **-W** flag is less than that for the **-C** flag. Otherwise, the **-C** flag will delete all the **C.*** (command) files before any warnings can be printed.

Cleaning Up Files for a Specific System

To delete files for one system, enter:

```
uucleanup  -shera
```

This removes all files using defaults for system `hera`, but does not remove any files for any other systems.

Files

/usr/sbin/uucp/*	Contains the **uudemon.cleanu** shell procedure and all the configuration files for BNU.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs for the **cron** daemon, including the **uudemon.cleanu** shell procedure.
/var/spool/uucp/*	Contain files removed by the **uucleanup** command.

Related Information

The **cron** daemon.

The **uucp** command, **uudemon.cleanu** command, **uustat** command, **uux** command, **uuclean** command.

BNU maintenance in *Networks and communication management*.

---

uucp Command

Purpose

Copies files from one system to another.

Syntax

uucp [**-c** | **-C**] [**-d** | **-f**] [**-g***Grade*] [**-j**] [**-m**] [**-n***User*] [**-r**] [**-s***File*] [**-x***DebugLevel*] *SourceFile ... DestinationFile ...*

Description

The **uucp** command is a Basic Networking Utilities (BNU) command that copies one or more source files from one system to one or more destination files on another UNIX system. Files can be copied within a local system, between a local and a remote system, and between two remote systems.

The **uucp** command accomplishes the file transfer in two steps: first, by creating a command (**C.***) file in the spooling directory on the local computer and then by calling the **uucico** daemon to send the request to the specified computer. Command files include information such as the full path name of the source and destination files and the sender's login name. The full path name of a command file is a form of the following:

/var/spool/uucp/*SystemName***/C.***SystemNameNxxxx*

where *N* is the grade of the request and *xxxx* is the hexadecimal sequence number used by BNU.

If the **uucp** command is used with the **-C** flag to copy the files to the spool directory for transfer, the **uucp** command creates not only a command file, but also a data (**D.***) file that contains the actual source file. The full path name of a data file is a form of the following:

/var/spool/uucp/*SystemName***/D.***SystemNamexxxx###*

Once the command files (and data files, if necessary) are created, the **uucp** command then calls the **uucico** daemon, which in turn attempts to contact the remote computer to deliver the files.

It is useful to issue the **uuname** command to determine the exact name of the remote system before issuing the **uucp** command. The **uulog** command provides information about **uucp** activities with another system.

Source and Destination File Names

File names and system names can contain only ASCII characters. Each can either be a path name on the local system or have the following form:

*SystemName***!***PathName*

where *SystemName* is taken from a list of system names that BNU knows about.

The destination *SystemName* can also be a list of names, such as the following:

*SystemName***!***SystemName***! . . . ! ***SystemName***!***PathName*

In this case, an attempt is made to send the file using the specified route to the destination. Make sure that intermediate nodes in this route are willing to forward information, and that they actually talk to the next system.

The shell pattern-matching characters ? (question mark), * (asterisk), and [**. . .**] (brackets and ellipsis) can be used in the path names of the source file; the appropriate system expands them. The shell pattern-matching characters should not be used in the path name of the destination file.

If the *DestinationFile* is a directory rather than a file, the **uucp** command uses the last part of the *SourceFile* name to name the transferred file on the remote system.

Path Names

Path names for the *SourceFile* and *DestinationFile* parameters contain only ASCII characters. Paths for the source file can be one of the following:

- A full path name
- A relative path name

Paths for the *DestinationFile* parameter can be in the forms for the *SourceFile* parameter, or can be one of the following:

- A path name preceded by *~User* (for example, `~jkimble`) where *User* is a login name on the remote system. The specified user's login directory is then considered the destination of the transfer. If the user specifies an invalid login name, the files are transferred to the public directory, **/var/spool/uucppublic**, which is the default.
- A path name preceded by *~/Destination*, where *Destination* is appended to **/var/spool/uucppublic**. The destination is treated as a file name unless more than one file is being transferred by the request, the destination already exists as a directory on the remote system, or the destination is specified as a directory.

  To specify the destination as a directory, follow the destination name with a / (slash). For example, `~/amy/` as the destination creates the directory `/var/spool/uucppublic/amy`, if it does not already exist, and puts the requested files in that directory.

Permissions

- The system administrator should restrict the access to local files by users on other systems.
- When transmitting files, the **uucp** command preserves execute permissions and grants read and write permissions to the owner, the group, and all others. (The **uucp** command owns the file.)
- Sending files to arbitrary *DestinationFile* path names on other systems or getting files from arbitrary *SourceFile* path names on other systems often fails because of security restrictions. The files specified in the path name must give read or write permission not only for the same group of users but also for any group.
- Protected files and files in protected directories owned by the requestor can be sent by the **uucp** command.

Flags

-c Prevents files from being copied. This flag is the default and should not be used with the **-C** flag. If both flags are specified, the **-c** flag is overridden.

-C Copies local files to the spool directory for transfer. Depending on the configuration of the **Poll** and **Systems** files and on how often the **uusched** daemon is run, the files may be transferred immediately on demand polling or in the future.

Occasionally, problems occur while transferring a source file; for example, the remote computer may not be working or the login attempt may fail. In such a case, the file remains in the spool directory until it is either transferred successfully or removed by a cleanup command.

This flag counteracts the **-c** flag.

-d Creates any intermediate directories needed to copy the source files to the destination files on a remote system. Instead of first creating a directory and then copying files to it, the **uucp** command can be entered with the destination path name, and the BNU Program will create the required directory. This flag is the default and cannot be used with the **-f** flag.

-f Does not create intermediate directories during the file transfer. This flag is used if the destination directory already exists and you do not want BNU to write over it. This command counteracts the **-d** flag.

-g*Grade* Specifies when the files are to be transmitted during a particular connection. The *Grade* variable is a single number (0 to 9) or letter (A to Z, a to z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest) grade. The default is **N**.

-j Displays the job identification number of the transfer operation on standard output. This job ID can be used with the **uustat** or **uuq** command to obtain the status of a particular job or with the **uustat -k** command or **uuq -d** command to terminate the transfer before it is completed.

-m	Sends a mail message to the requester when the source file is successfully copied to the destination file on a remote system. The message is sent to the requester's mailbox, **/var/spool/mail/***User*. The **mail** command does not send a message for a local transfer. The **-m** flag works only when sending files or receiving a single file. It does not work when forwarding files.
-n*User*	Notifies the recipient on the remote system identified by the *User* entry that a file has been sent. The mail system does not send a message for a local transfer. User names can contain only ASCII characters. Receiving multiple files specified by the shell pattern-matching characters ? (question mark), * (asterisk), and [. . .] (brackets and ellipses) does not activate the **-n** option.
-r	Prevents the starting of the **uucico** file transfer daemon, even if the command was issued at a time when calls to the remote system are permitted. (By default, a call to the remote system is attempted if the command is issued during a time period specified in the **Poll** and **Systems** files.) The **-r** option is useful for debugging.
-s*File*	Reports the status of the transfer to the specified file. In this case, the *File* variable must designate a full path name.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable is a number from 0 to 9. The higher the number, the more detailed the report.

Examples

1. To copy a file from the local system to a remote system, enter:
   ```
   uucp /home/geo/f1 hera!/home/geo/f1
   ```
   In this example, the `f1` file from the local system is copied to remote system `hera`.
2. To copy a file from the remote system and place it in the public directory, enter:
   ```
   uucp hera!geo/f2 /var/spool/uucppublic/f2
   ```
   In this example, the `f2` file from remote system `hera` is copied and placed in the public directory.
3. To copy a file from the remote system and place it in a directory other than the public directory, enter:
   ```
   uucp hera!geo/f2 /home/geo/f2
   ```
   In this example, the `f2` file from the remote system `hera` is copied to the `/home/geo/f2` directory. The `geo` login directory must allow write permission to members of the other group, for example, with mode 777.

Files

/usr/bin/uucp	Contains the **uucp** command.
/etc/uucp/Poll	File listing times when remote systems are automatically called (polled).
/etc/uucp/Systems	File describing accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems** files.
/var/spool/uucp	Spooling directory containing BNU status information.
/var/spool/uucppublic	Public directory containing files awaiting transfer by the **uucico** daemon.
/var/spool/uucppublic/*SystemName***/C.***	Contains command files.
/var/spool/uucppublic/*SystemName***/D.***	Contains data files.

Related Information

The **ct** command, **cu** command, **mail** command, **uuclean** command, **uucleanup** command, **uulog** command, **uuname** command, **uupick** command, **uuq** command, **uustat** command, **uuto** command, **uux** command.

The **uucico** daemon, **uusched** daemon.

---

uucpadm Command

Purpose

Enters basic BNU configuration information.

Syntax

uucpadm

Description

The **uucpadm** command provides interactive entry and modification of basic BNU configuration information in the **Devices**, **Systems**, **Permissions**, **Poll**, and **Dialcodes** files in the **/etc/uucp** directory. You can use the **uucpadm** command repeatedly to adjust the same file.

When you enter the **uucpadm** command at the command line, the command displays a list of the files you can change. After you choose a file to modify, the command displays a vertical list of the names of the fields in that file. You can enter the appropriate entry in each field. When you press the Enter key, the cursor moves to the next field in the list.

The command uses a copy of a file to record changes. The original file remains unchanged until you press the Ctrl+U or Ctrl+X key sequence at the appropriate menu. You can exit to the main **uucpadm** menu at any time, without saving your changes, by using the Ctrl+D key sequence.

The help routine provides instructions for each data field. Type a ? (question mark) in any menu field to access the help routine for that field.

Type a ~ (tilde) in any field to enter an ASCII editor and edit the appropriate file for that field. The **uucpadm** command invokes the editor designated by the **EDITOR** environment variable. If the **EDITOR** variable is not defined, the command invokes the **vi** editor.

If your entry for the first menu item matches an existing record, the **uucpadm** command retrieves that record for update. The command also tells you how many records have that first entry. If your entry for the first menu item does not match any existing record, the **uucpadm** command displays the word `ADD` at the top of the screen.

The **uucpadm** command checks the data as you enter it. If an inconsistency among the files is found, the command displays a warning message.

If the **uucpadm** command recognizes the entry you make for the first menu item, it fills in the default values for the remaining fields. For example, if you type `TCP` as the `Type` in the **Devices** file menu, the command places a - (hyphen) in each remaining field for you. It also checks for consistency with other files and for processes that should be running on the system. For example, when you type `TCP` as the `Type` in the **Devices** file menu, the **uucpadm** command checks to see if the **uucpd** daemon is running. If the daemon is not running, the command displays a note after the **Type** field, as follows:

```
Type: TCP
       <Note: Make certain uucpd is enabled.>
Line1: -
```

Note: The **uucpadm** command does not edit the **/etc/uucp/Dialers** file. Use an ASCII editor to edit this file.

Mode	File
rw	**/etc/uucp/Devices**
rw	**/etc/uucp/Dialcodes**
rw	**/etc/uucp/Permissions**
rw	**/etc/uucp/Poll**
rw	**/etc/uucp/Systems**

Examples

1. To start the **uucpadm** command, type the following:

   ```
   /usr/sbin/uucp/uucpadm
   ```

   A menu listing the files you can change is displayed.
2. To make an entry to the **/etc/uucp/Devices** file, choose the `Add/Change Uucp Devices` option at the **uucpadm** menu. The following is a sample **uucpadm** screen defining a `direct 9600 baud` connection to system `merlin` over the `tty3` device:

   ```
   Type: merlin
   line1: tty3
   line2: -
   class: 9600
   dialers: direct
   ```
3. To make an entry to the **/etc/uucp/Systems** file, choose the `Add/Change Uucp Systems` option at the **uucpadm** menu. The following is a sample **uucpadm** screen defining the `nostromo.aus.ibm.com` system connected to an `ACU` device in `class 2400`:

   ```
   Name: nostromo.aus.ibm.com
   Time: Any
   Type: ACU
   Class: 2400
   Phone: 997-7942
   Login: nuucp
   Password: gotcha
   ```
4. To change the **/etc/uucp/Permissions** file, choose the `Add/Change Uucp Permissions File` option at the **uucpadm** menu.
   a. Following is a sample **uucpadm** screen defining a LOGNAME entry in the **Permissions** file:

      ```
      L/M: LOGNAME=uucpz
      Request: yes
      Sendfiles: yes
      Read: /
      Write: NOWRITE=/etc
      Callback:
      Commands:
      Validate: merlin:nostromo
      ```

      If the remote machine is `merlin` or `nostromo`, the login ID must be `uucpz` (VALIDATE option). Remote hosts using this ID can request to send files, and the local host can sendfiles as requested. Users with this ID can read all files with permissions granted to the others group, and can write to all files, except those in the **/etc** directory, with permissions granted to the others group.

b. Following is a sample **uucpadm** screen defining a MACHINE entry in the **Permissions** file:

```
L/M: MACHINE=merlin
Request: yes
Sendfiles:
Read: NOREAD=/etc
Write: NOWRITE=/etc
Callback:
Commands: ALL
Validate:
```

The machine ID is `merlin`. Requests for file transfers can be made. The user can read all files and can write to all files except those in the **/etc** directory. The execution of all commands is permitted.

5. To make an entry in the **/etc/uucp/Poll** file, choose the `Add/Change Uucp Poll File` option at the **uucpadm** menu. Following is a sample **uucpadm** screen defining an entry in the **Poll** file:

```
System: merlin
Hours: 0 7 13 19
```

This entry instructs BNU to poll the `merlin.aus.ibm.com` system at 2400 hours (midnight), 700 hours (7 a.m.), 1300 hours (1 p.m.), and 1900 hours (7 p.m.).

6. To make an entry in the **/etc/uucp/Dialcodes** file, choose the `Add/Change Uucp Dialcodes` option at the **uucpadm** menu. Following is a sample **uucpadm** screen defining an entry in the **Dialcodes** file:

```
Abr: LA
Dialcode: 1-213-
```

This entry assigns `LA` as the abbreviation for the Los Angeles area code.

Files

/usr/sbin/uucp/uucpadm	Contains the **uucpadm** command.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Poll	Specifies when BNU polls remote systems to initiate tasks.
/etc/uucp/Systems	Describes accessible remote systems.

Related Information

The **uucp** command, **uuname** command.

The **uucheck** command checks the **/etc/uucp/Permissions** file for correct configuration.

BNU configuration for a telephone connection example, Example of BNU configuration for a TCP/IP connection, and BNU configuration for a direct connection in *Networks and communication management*.

BNU file and directory structure in *Networks and communication management*.

Configuring BNU in *Networks and communication management*.

---

uucpd Daemon

Purpose

Handles communications between BNU and TCP/IP.

Syntax

The **uucpd** daemon cannot be started from the command line. It is started by the **inetd** daemon.

uucpd

Description

The **uucpd** daemon is an internal program that enables users of systems linked by the Basic Networking Utilities (BNU) program to establish a TCP/IP connection to other systems linked over a Token-Ring, Ethernet, or other network.

The **uucpd** daemon is a subserver of the **inetd** daemon. The **uucpd** daemon must be running as a background process on all the networked systems before the BNU program can use TCP/IP system to communicate. If the **uucpd** daemon is not running, reconfigure the **inetd** daemon to start the **uucpd** daemon. Use the **netstat** command to find out if the **uucpd** daemon is running.

Files

/etc/hosts	Contains the host name table used by TCP/IP.
/etc/inetd.conf	Contains the configuration of the **inetd** daemon.
/etc/services file	Defines socket assignments used by TCP/IP.
/usr/sbin /uucpd	Contains the **uucpd** daemon.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.

Related Information

The **inetd** daemon.

Configuring the inetd daemon in *Networks and communication management*.

Transmission Control Protocol/Internet Protocol in *Networks and communication management*.

BNU daemons, Configuring BNU in *Networks and communication management*.

---

uudecode Command

Purpose

Decodes a binary file that was used for transmission using electronic mail.

Syntax

uudecode [**-o** *OutputFile*] [*InFile*]

Description

The **uudecode** command reads an encoded file, strips off leading and trailing lines added by mailers, and recreates the original file with the specified mode and name. Decoding a file causes the result to be automatically saved to a file. The file name is identical to the remote file argument originally supplied to the **uuencode** command unless an output file name is specified with the **-o** flag.

Flags

-o *OutputFile* Specifies the output file name that will be used instead of any pathname contained in the input data. You can direct the output of **uudecode** to standard output by specifying **/dev/stdout** as the *OutputFile*.

Parameters

InFile Specifies the name of the file to decode.

Example

To decode the file `/tmp/con` on a local system that was encoded with the follwing command:

```
uuencode /usr/lib/boot/unix pigmy.goat > /tmp/con
```

enter:

```
uudecode /tmp/con
```

The file `pigmy.goat` will be identical to the originally encoded file `/usr/lib/boot/unix`.

Files

/usr/bin/uudecode Contains the **uudecode** command.

Related Information

The **mail** command, **rmail** command, **sendmail** command, **uucp** command, **uuencode** command, **uusend** command, **uux** command.

---

uudemon.admin Command

Purpose

Provides periodic information on the status of BNU file transfers.

Syntax

uudemon.admin

Description

The **/usr/sbin/uucp/uudemon.admin** command is a shell procedure that mails status information about the Basic Networking Utilities (BNU) activities to the **uucp** login ID at intervals specified in the **/var/spool/cron/crontabs/uucp** file. The command executes both the **uustat -p** and the **uustat -q** commands:

- The **-p** flag instructs the **uustat** command to run the **ps -flp** command (process status, which generates a full, long list of specified process IDs) for all process ID (PID) numbers in the lock files.
- The **-q** flag lists the jobs currently queued to run on each system. These jobs either are waiting to execute or are in the process of executing. If a status file exists for the system, its date, time, and status information are reported.

Execute the **uudemon.admin** command at least once a day. The **uudemon.admin** command is not enabled when you install the BNU program. To run this command automatically, edit the

/var/spool/cron/crontabs/uucp file, removing the comment character (#) from the beginning of the line that governs running the **uudemon.admin** command.

Examples

To run the **uudemon.admin** command automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (#) from the beginning of the **uudemon.admin** command line. Change:

```
#48 8,12,16 * * * /usr/bin/sh -c
 "/usr/sbin/uucp/uudemon.admin > /dev/null"
```

to:

```
48 8, 12, 16 * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.admin > /dev/null"
```

The `48` notation represents minutes, the `8,12,16` notation represents hours based on the 24-hour clock, and the three asterisks (`* * *`) are placeholders representing the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the **uudemon.admin** command daily at 48 minutes past the hours 0800, 1200, and 1600, that is, at 8:48 a.m., 12:48 p.m., and 4:48 p.m., respectively.

Note: These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.admin** command to fit the needs of your site.

Files

/usr/sbin/uucp/uudemon.admin	Contains the **uudemon.admin** command and the configuration files for BNU.
/etc/locks/*	Contains lock files which prevent multiple uses of devices and multiple calls to systems.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.admin** command, for the **cron** daemon.

Related Information

The **uustat** command.

BNU maintenance commands in *Networks and communication management*.

---

uudemon.cleanu Command

Purpose

Cleans up BNU spooling directories and log files.

Syntax

uudemon.cleanu

Description

The **/usr/sbin/uucp/uudemon.cleanu** command is a shell script that cleans up the Basic Networking Utilities (BNU) spooling directories and log files. The command deletes files in the spooling directories that are as old as, or older than, a specified number of days, and then removes empty spooling directories.

The **uudemon.cleanu** command also updates archived log files by removing log information more than three days old. The command removes log files for individual computers from the **var/spool/uucp/.Log** directory, merges them, and places them in the **var/spool/uucp/.Old** directory, which contains old log information.

After performing the cleanup operations, the **uudemon.cleanu** command mails the **uucp** login ID a summary of the status information gathered during the current day.

Instruct the **cron** daemon to run the **uudemon.cleanu** command daily, weekly, or at longer intervals, depending on the amount of transactions the **uucico** and **uuxqt** daemons perform on the local system.

To run this command automatically, remove the comment character (#) at the beginning of the **uudemon.cleanu** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.cleanu** command is not usually entered on the command line but is instead executed by the **cron** daemon.

Example

To run the **uudemon.cleanu** procedure automatically, edit the **/var/spool/cron/crontabs/uucp** file and uncomment the **uudemon.cleanu** line. Change:

```
# 45 23 * * * /usr/bin/sh -c
 "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
```

to:

```
45 23 * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
```

The 45 notation represents minutes, the 23 notation represents hours based on the 24-hour clock, and the three asterisks (* * *) are placeholders representing the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the **uudemon.cleanu** shell procedure at 45 minutes after hour 2300-that is, at 11:45 p.m.

Notes:

1. These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.cleanu** command so that they fit the needs of your site.
2. The system allots the BNU program a specified amount of storage space for any one particular log file; the number of blocks is determined by the default **ulimit** value. If the **uudemon.cleanu** command fails to execute because the **ulimit** value is set too low for the requirements of the local system, delete the **uudemon.cleanu** command line (shown previously) from the **/var/spool/cron/crontabs/uucp** file and add the following entry to the **root** crontabs file, **/var/spool/cron/crontabs/root**:

   ```
   45 23 * * * ulimit 5000; /usr/bin/su uucp
    -c "/usr/sbin/uucp/uudemon.cleanu > /dev/null"
   ```

   Put the text on one line when entering it in the **root** crontabs file.

Files

/usr/sbin/uucp/uudemon.cleanu	Contains the **uudemon.cleanu** command.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.cleanu** command, for the **cron** daemon.
/var/spool/cron/crontabs/root	Schedules root user jobs for the **cron** daemon.
/var/spool/uucp/.Log /*	Contains the BNU program log files.

Related Information

The **uustat** command, **uux** command.

The **cron** daemon, **uucico** daemon, **uuxqt** daemon.

BNU log files in *Networks and communication management*.

uudemon.hour Command

Purpose

Initiates file transport calls to remote systems using the BNU program.

Syntax

uudemon.hour

Description

The **/usr/sbin/uucp/uudemon.hour** command is a shell procedure used by the Basic Networking Utilities (BNU). In conjunction with the **Poll** file, the **uudemon.poll** command, and the **/var/spool/cron/crontabs/uucp** file, the **uudemon.hour** command initiates calls to remote systems.

The **uudemon.hour** command calls the following programs, which are involved in transferring files between systems at specified hourly intervals:

- The **uusched** daemon first searches the spooling directory on the local system for command files that have not been transferred to the specified remote system, and then schedules the transfer of those files.
- The **uuxqt** daemon searches the spooling directory for execute files that have been transferred to the local system but have not yet been processed on that system.

Instruct the **cron** daemon to run the **uudemon.hour** command at specified hourly intervals. The frequency at which you run the **uudemon.hour** command depends on the amount of file-transfer activity originating from the local computer. If users on the local system initiate a large number of file transfers, you may need to specify that the **cron** daemon should start the **uudemon.hour** command several times an hour. If the number of file transfers originating from the local system is low, you can probably specify a start time once every 4 hours, for example.

To run the **uudemon.hour** command automatically, remove the comment character (#) from the beginning of the **uudemon.hour** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.hour** command is not usually entered on the command line, but is executed by the **cron** daemon.

Example

To run the **uudemon.hour** command automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the comment character (#) at beginning of the **uudemon.hour** command line. Change:

```
#25,55 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.hour >  /dev/null"
```

to:

```
25,55 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.hour >  /dev/null"
```

The `25,55` notation represents minutes, and the four asterisks (`* * * *`) are placeholders representing the hour of the day, the day of the month, the month of the year, and the day of the week, respectively. Therefore, this line instructs the **cron** daemon to run the **uudemon.hour** command at 25 minutes past the hour and again at 55 minutes past the hour; for example, at 8:25 and 8:55 a.m., again at 9:25 and 9:55 a.m., and again every hour of every day.

Notes:

1. These run intervals are defaults. By altering them, you can change the times at which the **cron** daemon executes the **uudemon.hour** command to fit the needs of your site. For example, to run the **uudemon.hour** command once every 4 hours, type the numeral `4` in the **time-interval** field.

2. If you change the run times for the **uudemon.hour** command, you should also change the run times for the **uudemon.poll** command so that it polls remote systems 5 to 10 minutes before the **uudemon.hour** command is run.

Files

/usr/sbin/uucp/uudemon.hour	Contains the **uudemon.hour** command.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.hour** and **uudemon.poll** commands, for the **cron** daemon.

Related Information

The **uudemon.poll** command.

The **cron** daemon, **uusched** daemon, **uuxqt** daemon.

Setting up BNU polling of remote systems in *Networks and communication management*.

BNU maintenance commands, BNU daemons in *Networks and communication management*.

---

uudemon.poll Command

Purpose

Polls the systems listed in the BNU **Poll** file.

Syntax

uudemon.poll

Description

The **/usr/sbin/uucp/uudemon.poll** command is a shell procedure used by the Basic Networking Utilities (BNU). In conjunction with the **/etc/uucp/Poll** file, the **uudemon.hour** command, and the **/var/spool/cron/crontabs/uucp** file, the **uudemon.poll** command initiates calls to remote systems.

The **uudemon.poll** command performs the following actions:

- Polls (contacts) the systems listed in the **Poll** file (**/etc/uucp/Poll**).
- Creates command (**C.***) files for the systems listed in the **Poll** file.

The time at which you run the **uudemon.poll** command depends on the time at which you run the **uudemon.hour** command. In general, schedule the polling shell procedure before the hourly procedure. This schedule enables the **uudemon.poll** command to create any required command files before the **cron** daemon runs the **uudemon.hour** command.

Instruct the **cron** daemon to run the **uudemon.poll** command about 5 to 10 minutes before running the **uudemon.hour** command. To run this procedure automatically, remove the comment character (#) from the beginning of the **uudemon.poll** command line in the **/var/spool/cron/crontabs/uucp** file.

Note: The **uudemon.poll** command is not usually entered on the command line, but is executed by the **cron** daemon.

Example

To run the **uudemon.poll** shell procedure automatically, edit the **/var/spool/cron/crontabs/uucp** file and remove the # (comment character) at the beginning of the line which starts the **uudemon.poll** command. Change:

```
#20,50 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.poll >  /dev/null"
```

to:

```
20,50 * * * * /usr/bin/sh -c "/usr/sbin/uucp/uudemon.poll >  /dev/null"
```

The `20,50` notation represents minutes, and the four asterisks (`* * * *`) are placeholders representing the hour of the day, the day of the month, the month of the year, and the day of the week, respectively. This line therefore instructs the **cron** daemon to run the **uudemon.poll** command at 20 minutes past the hour and again at 50 minutes past the hour-for example, at 8:20 and 8:50 a.m., and at 9:20 and 9:50 a.m.-every hour of every day.

Note: Change the times at which the **cron** daemon executes the **uudemon.poll** command to correspond to the times you set up for the **uudemon.hour** command. The defaults specified in the **/var/spool/cron/crontabs/uucp** file instruct the **cron** daemon to run the **uudemon.poll** command 5 minutes before running the **uudemon.hour** command.

Files

/usr/sbin/uucp/*	Contains the **uudemon.poll** and **uudemon.hour** commands and all the configuration files for BNU.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs, including the **uudemon.poll** command, for the **cron** daemon.

Related Information

The **uudemon.hour** command.

The **cron** daemon.

Setting up BNU polling for remote systems in *Networks and communication management*.

BNU maintenance commands, BNU daemons in *Networks and communication management*.

---

uuencode Command

Purpose

Encodes a binary file for transmission using electronic mail.

Syntax

uuencode [**-m**] [*SourceFile*] *OutputFile*

Description

The **uuencode** command converts a binary file to ASCII data. This is useful before using BNU (or uucp) mail to send the file to a remote system. The **uudecode** command converts ASCII data created by the **uuencode** command back into its original binary form.

The **uuencode** command takes the named *SourceFile* (default standard input) and produces an encoded version on the standard output. The encoding uses only printable ASCII characters, and includes the mode of the file and the *OutputFile* filename used for recreation of the binary image on the remote system.

Use the **uudecode** command to decode the file.

Flags

-m	Encode the output using the MIME Base64 algorithm. If **-m** is not specified, the old **uuencode** algorithm will be used.

Parameters

OutputFile	Specifies the name of the decoded file. You can direct the output of the **uuencode** command to standard output by specifying **/dev/stdout** as the *OutputFile*.
SourceFile	Specifies the name of the binary file to convert. Default is standard input.

Examples

1. To encode the file `unix` on the local system and mail it to the user `jsmith` on another system called `mysys`, enter:

   ```
   uuencode unix unix | mail jsmith@mysys
   ```

2. To encode the file `/usr/lib/boot/unix` on your local system with the name `pigmy.goat` in the file `/tmp/con` , enter:

   ```
   uuencode /usr/lib/boot/unix pigmy.goat > /tmp/con
   ```

Files

/usr/bin/uuencode	Contains the **uuencode** command.

Related Information

The **mail** command, **rmail** command, **sendmail** command, **uucp** command, **uudecode** command, **uusend** command, **uux** command.

---

uuid_gen Command (NCS)

Purpose

Generates Universal Unique Identifiers (UUIDs) for objects, types, and interfaces.

Syntax

/etc/ncs/uuid_gen [**-c**] [**-p**] [**-C**] [**-P**]

Description

The **uuid_gen** program generates Universal Unique Identifiers (UUIDs). By default, it generates a character-string representation of a UUID. The options for **uuid_gen** enable you to generate source-code representations of UUIDs, suitable for initializing variables of type **uuid_$t**.

Flags

-C Generates a C source-code representation of a UUID.
-c Generates a template, including a UUID attribute, for an interface definition in the C syntax.
-P Generates a Pascal source-code representation of a UUID.
-p Generates a template, including a UUID attribute, for an interface definition in the Pascal syntax.

Examples

1. To generate a character-string representation of a UUID, enter the following:

   ```
   /etc/ncs/uuid_gen
   ```

   This produces the following output:

   ```
   34dc23469000.0d.00.00.7c.5f.00.00.00
   ```

2. To generate a template for an interface definition in the C syntax, enter:

   ```
   /etc/ncs/uuid_gen -c
   ```

   This produces the following output:

   ```
   %c
   [
   uuid(34dc239ec000.0d.00.00.7c.5f.00.00.00),
   version(1)
   ]
   interface INTERFACENAME {

   }
   ```

3. To generate a C source-code representation of a UUID, enter the following:

   ```
   /etc/ncs/uuid_gen -C
   ```

   This produces the following output:

   ```
   = { 0x34dc23af,
       0xf000,
       0x0000,
       0x0d,
       {0x00, 0x00, 0x7c, 0x5f, 0x00, 0x00, 0x00} };
   ```

4. To generate a template for an interface definition in the Pascal syntax, enter the following:

   ```
   /etc/ncs/uuid_gen -p
   ```

   This produces the following output:

   ```
   %pascal
   [
   uuid (458487b55160.02.c0.64.02.03.00.00.00),
   version (1)
   ]
   interface INTERFACENAME;

   end;
   ```

5. To generate a Pascal source-code representation of a UUID, enter the following:

   ```
   /etc/ncs/uuid_gen -P
   ```

   This produces the following output:

   ```
   := [
       time_high := 16#458487df,
       time_low := 16#9fb2,
       reserved := 16#000,
   ```

```
family := chr(16#02),
host := [chr(16#c0), chr(16#64), chr(16#02), chr(16#03),
         chr(16#00), chr(16#00), chr(16#00)]
]
```

Related Information

List of NCS Commands.

---

uukick Command

Purpose

Uses debugging mode to contact a specified remote system.

Syntax

uukick [**-x***DebugLevel*] *SystemName*

Description

The **uukick** command contacts a remote system, named by the *SystemName* parameter, using debugging mode. The debugging mode provides a means of monitoring Basic Networking Utilities (BNU) file transfers and connections to remote computers.

The **uukick** command starts the **uucico** daemon, which actually contacts the specified remote system. The **uucico** daemon produces debugging output that enables you to monitor its progress as it establishes the connection to the remote system, performs the remote login, and transfers a file.

The debugging output is scrolled on the screen of the local system. Once the system has finished displaying this information, press the Interrupt key to return to the prompt.

Notes:

1. Either you must be in the **/usr/lib/uucp** directory when you issue the **uukick** command, or you must issue the command with the full path name, **/usr/sbin/uucp/uukick**.
2. The **uukick** command is a shell script stored in the **/usr/lib/uucp** directory.

Flags

-x*DebugLevel* Overrides the default amount of detail in the debugging information the command displays on the screen. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the final report to be more detailed. If the **-x** flag is not used, the **uucico** daemon is started with the default level, which produces a moderate amount of information.

Example

To change the amount of detail in the information about the progress of the operation of the **uucico** daemon, use the **-x** flag to specify a higher or lower debugging level. For example, enter:

```
uukick  -x9 hera
```

This instructs the **uukick** command to generate as much information as possible about the way in which the **uucico** daemon is working while trying to connect to system `hera`. Or, enter:

```
uukick  -x3 hera
```

This instructs the command to generate less than the default amount of information about the connection.

Files

/usr/sbin/uucp/uukick	Contains the **uukick** shell script.
/etc/uucp	Contains the configuration files for BNU.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dialing code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/*	Contain files to be transferred and files recording transfer statistics.
/var/spool/uucppublic/*	Contain files that have been transferred.

Related Information

The **tail** command, **uucp** command, **uutry** command, **Uutry** command, **uux** command.

The **uucico** daemon, **uucpd** daemon.

Monitoring a BNU remote connection and Monitoring a BNU file transfer in *Networks and communication management*.

Maintaining BNU in *Networks and communication management*.

---

uulog Command

Purpose

Provides information about BNU file-transfer activities on a system.

Syntax

uulog [**-x**] [**-***Number*] [**-f***System* | **-s***System*]

Description

The Basic Networking Utilities (BNU) **uulog** command displays the contents of the log files containing the activities of the **uucico** and **uuxqt** daemons. Individual log files are created for each remote system with which the local system uses the **uucp**, **uuto**, and **uux** commands to communicate.

Use the **uulog** command to display a summary of **uucp**, **uuto**, and **uux** command requests by the user or by the system. All of these transactions are logged in files in the **/var/spool/uucp/.Log** directory. The files are named *DaemonName*/*SystemName* where the *DaemonName* directory is named for the daemon involved and the *SystemName* file is named for the remote system the daemon is contacting.

The **uucp** and **uuto** commands call the **uucico** daemon. The **uucico** daemon's activities are logged in the *SystemName* file in the **/var/spool/uucp/.Log/uucico** directory.

The **uux** command calls the **uuxqt** daemon. The **uuxqt** activities are logged in the *SystemName* file in the **/var/spool/uucp/.Log/uuxqt** directory.

You can examine these individual log files by issuing the **uulog** command directly. However, you can also have the BNU program automatically append these temporary log files to a primary log file that you can then examine. This is called *compacting the log files* and is handled by the **uudemon.cleanu** command, a shell script.

Flags

-f*System* Issues a **tail** command with the **-f** flag on the file transfer log for the specified *System* variable, displaying the end of the log file. Press the Interrupt key to leave the file and return to the prompt.

-s*System* Displays a summary of copy (**uucico** daemon) requests involving the specified system.

Notes:

1. System names can contain only ASCII characters.
2. The **-f** and **-s** flags cannot be combined.

-x Displays the **uuxqt** daemon log file for the given system.

-*Number* Displays the last lines of the file. The number of lines is determined by the *Number* variable. (To display the lines, the **uulog** command issues a **tail** command with the **-f** flag for the specified number of lines.)

Examples

1. To display the **uucico** log file for system `hera`, enter:

   ```
   uulog  -shera
   ```

   The output from the command is similar to the following:

   ```
   uucp hera (10/30-10:18:38,3833,0) SUCCEEDED (call to  hera)
   uucp hera (10/30-10:18:39,3833,0) OK (startup)
   jim hera heraN661d (10/30-10:18:39,3833,0) REQUEST
   (nostromo!D.hera661e6c9 --> hera!X.heraN661d (jim))
   jim hera heraN661d (10/30-10:18:40,3833,0) FAILED (CAN'T
   READ /var/spool/uucp/hera/D.hera661e6c9 13)
   uucp hera (10/30-10:18:41,3833,0) OK (conversation
   complete -8)
   ```

   The preceding lines log a conversation between the local system (`nostromo`) and remote system `hera`. The conversation began at `10:18:38` (a.m.) on October 30th, and ended at `10:18:41`. User `jim` attempted to transfer a data file, `D.hera661e6c9`, to system `hera`. The connection to `hera` was successful, but the file could not be transferred because BNU could not read it.

2. To display the **uuxqt** log file, enter:

   ```
   uulog  -x
   ```

3. To display the last forty lines of the file transfer log for system `zeus`, enter:

   ```
   uulog  -fzeus  -40
   ```

Files

/usr/bin/uulog	Contains the **uulog** command.
/var/spool/uucp/.Log	Contain the BNU log files.

Related Information

The **tail** command, **uucp** command, **uudemon.cleanu** command, **uuto** command, **uux** command.

The **uucico** daemon, **uuxqt** daemon.

BNU log files in *Networks and communication management*.

uuname Command

Purpose

Provides information about other systems accessible to the local system.

Syntax

uuname [**-c** | **-l**]

Description

The **uuname** command is a Basic Networking Utilities (BNU) command that displays a list of all the computers networked to the local system. This list of accessible systems is displayed on the screen of the local terminal.

In order for a local system to communicate with a remote system by way of BNU, the remote system must:

- Have a UNIX-based operating system.
- Be connected to the local system. (A telephone line can serve as the connection media.)

BNU can be used to communicate between a workstation and a non-UNIX-based operating system, but such communications may require additional hardware or software. The remote systems accessible with BNU commands are identified when the BNU programs are installed and listed in a BNU **Systems** file (by default, the **/etc/uucp/Systems** file, or one or more files specified in the **/etc/uucp/Sysfiles** file).

Before copying a file to another system with the **uuto** or **uucp** command, issue the **uuname** command to determine the exact name of the remote system.

Flags

-c Displays only the names of systems contained in the **cu Systems** files (configured by the **/etc/uucp/Sysfiles** file). Omission of this flag displays the names of systems contained in the **uucico Systems** files (also configured by the **/etc/uucp/Sysfiles** file). If **/etc/uucp/Sysfiles** is not used to separate **cu** and **uucico** configuration into separate **Systems** files, the names of all systems listed in **/etc/uucp/Systems** are displayed regardless of the **-c** flag.

-l Displays the name of the local system.

Examples

1. To identify the remote systems connected to the local system, enter:

   ```
   uuname
   ```

   The system responds with a list similar to the following:

   ```
   arthur
   hera
   merlin
   zeus
   ```

2. To identify the name of the local system, enter:

   ```
   uuname -l
   ```

   The system responds with something similar to the following:

   ```
   nostromo
   ```

Files

/usr/bin/uuname	Contains the **uuname** command.
/etc/uucp/Systems	Lists accessible remote systems.
/etc/uucp/Sysfiles	Specifies alternate files to be used as **Systems** files.
/var/spool/uucp	Contains BNU administrative files.
/var/spool/uucppublic	Contains BNU files awaiting transfer (public directory).

Related Information

The **ct** command, **cu** command, **uname** command, **uucp** command, **uupick** command, **uustat** command, **uuto** command, **uux** command.

---

uupick Command

Purpose

Completes the transfer of and handles files sent by the **uuto** command.

Syntax

uupick [**-s***System*]

Description

The **uupick** command is a Basic Networking Utilities (BNU) command that completes the transfer and handles files that the BNU **uuto** command has transmitted to a designated user ID.

Once the copied file is the receive directory, the **rmail** command notifies the recipient that the file has arrived. The recipient then issues the **uupick** command, which searches the public directory on the local system for files sent with some form of the following name:

/var/spool/uucppublic/receive/*User*/*System*/*File*

For each file or directory found, the **uupick** command displays the following message on the screen of the local system:

```
from System: [file File] [dir Directory]
?
```

The question mark prompt (`?`) following the message indicates you can now enter one of the file-handling options.

Flags

-s *System*	Searches **/var/spool/uucppublic/receive/***User*/*System* for files sent from the specified system. System names contain only ASCII characters.

File-Handling Options

The question mark prompt (`?`) following a message indicates that one of the following file-handling options should be entered:

Option	Action
!*Command*	Escapes to a shell to run the specified command. After the command executes, the user is automatically returned to the **uupick** command.
*	Displays all the file-handling options.

Option	Action
a [*Directory*]	Moves all **uuto** files currently in the **receive** directory into a specified directory on the local system. The default is the current working directory. Use a full or relative path name to specify the destination directory.
Ctrl-D	Stops processing and exits from the **uupick** command.
d	Deletes the specified file.
m [*Directory*]	Moves the file to a specified directory. If the *Directory* variable is not specified as a complete path name, a destination relative to the current directory is assumed. If no destination is given, the default is the current working directory on the local system.
new-line	Moves to the next entry in the receive directory when the Enter key is pressed.
p	Displays the contents of the file on the workstation screen.
q	Stops processing and exits from the **uupick** command.

Examples

1. To receive a file sent with the **uuto** command and add it to the current working directory, enter:

   ```
   uupick
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file1
   ?
   ```

   Enter:

   ```
    a
   ```

   In this example, the `/usr/bin/file1` file sent with the **uuto** command from system `anchor` is added to the current working directory.
2. To receive a file sent with the **uuto** command and add it to a specified directory on your local system, enter:

   ```
   uupick
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file2
   ?
   ```

   Enter:

   ```
    a /usr/bin1
   ```

   In this example, the `/usr/bin/file2` file sent with the **uuto** command from system `anchor` is added to the `/usr/bin1` directory on the local system.

   > **Note:** The `a /usr/bin1` instruction means move *all* files, not just one. Thus, if any other files are in the `~/anchor/...` directory, they will also be moved.
3. To search for files sent from system `anchor`, enter:

   ```
   uupick  -s anchor
   ```

   The system responds with a message similar to:

   ```
   from system anchor: file file1
   ```

Files

/usr/bin/uupick	Contains the **uupick** command.
/var/spool/uucppublic	Contains the BNU public directory.

Related Information

The **ct** command, **cu** command, **uucp** command, **uuname** command, **uustat** command, **uuto** command, **uux** command.

---

uupoll Command

Purpose

Forces a poll of a remote BNU system.

Syntax

uupoll [**-g***Grade*] [**-n**] *SystemName*

Description

The **uupoll** command forces the Basic Networking Utilities (BNU) to poll the remote system specified by the *SystemName* parameter. The command is usually run by the **cron** daemon or by a user who wants to force a job to be executed immediately. Otherwise, remote systems are polled by the **uudemon.poll** command at times scheduled in the **/etc/uucp/Poll** file and the **/var/spool/cron/crontabs/uucp** file.

Normally, the **uucico** daemon contacts a remote system only at times specified in the **Poll** file or when there is a job queued for that system. The **uupoll** command queues a null job for the remote system and then invokes the **uucico** daemon. This forces the **uucico** daemon to contact the remote system immediately and attempt to send any jobs which are queued for that system. Use the **-g** flag to specify that only high priority jobs be sent.

Use the **-n** flag to queue the null job without starting the **uucico** daemon. Use this option to:

- Queue a null job before invoking the **uucico** daemon for debugging.
- Queue a null job just before the **uucico** daemon is usually invoked, thus forcing the daemon to poll the specified system.

The *SystemName* parameter is required, and specifies the name of the remote system to be polled.

Flags

-g*Grade*	Instructs the **uupoll** command to send only jobs of the given grade (specified by the *Grade* parameter) or higher on this call. Jobs of a lower grade will remain in the queue until the next time the remote system is polled.
-n	Queues the null job, but does not invoke the **uucico** daemon.

Examples

1. To run the **uupoll** command with the **cron** daemon, place an entry in your **crontabs** file similar to:

   ```
   0 1,7,16 * * * /usr/bin/uupoll hera
   ```

   This polls system `hera` at 0100 hours (1 a.m.), 0700 hours (7 a.m.), and 1600 hours (4 p.m.) daily.

2. If the local system already runs the **uucico** daemon at specific times, you may want to queue a null job just before the **uucico** daemon normally runs. For example, if your system runs the **uucico** daemon hourly, place an entry similar to the following in your **crontabs** file:

```
0  1,7,16  *  *  *  /usr/bin/uupoll  -n zeus
0  5,12,21 *  *  *  /usr/bin/uupoll -n hera
5  *       *  *  *  /usr/sbin/uucp/uucico -r1
```

   This queues null jobs for the remote sites on the hour, and they are processed by the **uucico** daemon when it runs at 5 minutes past the hour.
3. To force the **uucico** daemon to transfer all jobs of grade `N` or higher for system `zeus`:

```
uupoll  -gN zeus
```

Files

/usr/bin/uupoll	Contains the **uupoll** command.
/etc/uucp/Poll	Specifies when the BNU program should poll remote systems to initiate tasks.
/var/spool/cron/crontabs/uucp	Schedules automatic polling of remote systems.
/var/spool/uucp/*SystemName*	Contain files to be transferred to remote systems.

Related Information

The **uucp** command, **uux** command.

The **Uutry** command invokes the **uucico** daemon with debugging turned on.

The **uudemon.poll** and **uudemon.hour** commands perform automatic polling of remote systems as scheduled by the **cron** daemon.

The **uucico** daemon.

Understanding the BNU Daemons in *Networks and communication management*.

---

uuq Command

Purpose

Displays the BNU job queue and deletes specified jobs from the queue.

Syntax

uuq [**-l** | **-h**] [**-s***SystemName*] [**-u***User*] [**-d***JobNumber*] [**-r***SpoolDir*] [**-b***BaudRate*]

Note: Only a user with root authority can use the **-d** flag.

Description

The **uuq** command is used to list or delete job entries in the Basic Networking Utilities (BNU) job queue.

When listing jobs, the **uuq** command uses a format similar to that used by the **ls** command. In the default format, the **uuq** command lists only the job numbers of the jobs waiting in the queue, followed by a summary line for each system.

In summary format (**uuq -h**) only the summary lines are listed. Summary lines give:

- System name
- Number of jobs for the system
- Total number of bytes to send

In the long format (**uuq -l**), which can be quite slow, the information listed for each job is:

- Job number
- Number of files to transfer
- User who sent the job
- Number of bytes to be sent
- Type of job requested:

S Sending a file
R Receiving a file
X Executing a command on the remote system

- File to be sent or received or the command to be executed

A user with root authority can use the **-d***JobNumber* flag to delete jobs from the queue after running a **uuq** listing to discover the job numbers.

Flags

-b*BaudRate*	Uses the baud rate given, instead of the default (1200 baud), to compute the transfer time.
-d *JobNumber*	Deletes the job designated by the *JobNumber* variable from the BNU queue. Only someone with root authority can delete jobs from the queue.
-h	Shows only the summary lines for each system.
-l	Lists the output in the long format.
-s*SystemName*	Lists only jobs for systems whose system names begin with the string specified in the *SystemName* variable.
-r *SpoolDir*	Searches for files in the spooling directory designated by the *SpoolDir* variable, instead of in the default spooling directory.
-u*User*	Lists only jobs queued by users whose login names begin with the string specified in the *User* variable.

Examples

1. To get a long listing of all jobs spooled for system `hera`, type:

   ```
   uuq -l -shera
   ```

2. To get a summary listing for all systems, type:

   ```
   uuq -h
   ```

3. To delete a job for user `nita` from the queue, first use the **uuq** command to find the number of the job you want to delete, as follows:

   ```
   uuq -l -unita
   ```

   This produces a list of jobs spooled for user `nita`. Find the job you wish to remove. If its job number is `13451`, for example, the following command will delete the job:

   ```
   uuq -d13451
   ```

Note: You must have root authority or be logged in as **uucp** to delete jobs from the queue.

Files

/usr/bin/uuq	Contains the **uuq** command.
/var/spool/uucp/*SystemName*	Contains spool files for the remote system designated by *SystemName*.
/var/spool/uucp/*SystemName***/C.***	Contain instructions for file transfers.
/var/spool/uucp/*SystemName***/D.***	Contain information about data files to be transferred.
/var/spool/uucp/*SystemName***/X.***	Contain instructions for executing remote commands.

Related Information

The **uucp** command, **uux** command, **uulog** command, **uusnap** command.

BNU daemons, BNU maintenance commands in *Networks and communication management*.

---

uusched Daemon

Purpose

Schedules work for the Basic Networking Utilities (BNU) file transport program.

Syntax

uusched [**-u***DebugLevel*] [**-x***DebugLevel*]

Description

The **uusched** daemon schedules work for the Basic Networking Utilities (BNU) file transport program. It schedules the transfer of files that are queued in the **/var/spool/uucp/***SystemName* directory. The scheduling daemon first randomizes the work and then starts the **uucico** daemon, which transfers the files.

The **uusched** daemon is usually started by the **uudemon.hour** command, a shell procedure, which is run periodically by the **cron** daemon based on instructions from the **/var/spool/cron/crontabs/uucp** file.

The **uusched** daemon can also be started from the command line for debugging purposes.

> **Note:** Either you must be in the **/usr/sbin/uucp** directory when you start the **uusched** daemon, or you must start the daemon with the full path name, **/usr/sbin/uucp/uusched.**

Flags

-u*DebugLevel*	Passes as the **-x***DebugLevel* flag to the **uucico** daemon. The *DebugLevel* variable is a number from 0 to 9, with a default of 5. Higher numbers give more detailed debugging information, which is displayed on the screen of the local system.
-x*DebugLevel*	Outputs debugging messages from the **uusched** daemon. The *DebugLevel* variable is a number from 0 to 9, with a default of 5. Higher numbers give more detailed debugging information, which is displayed on the screen of the local system.

Example

To start the **uusched** daemon from the command line, enter:

```
/usr/sbin/uucp/uusched &
```

This starts the **uusched** daemon as a background process. (Note that the path name is included in the command.)

Files

/etc/locks /*	Contains lock files that prevent multiple uses of devices and multiple calls to systems.
/usr/sbin/uucp/*	Contains the **uusched** daemon and the BNU configuration files.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Maxuuscheds	Limits scheduled jobs.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/cron/crontabs/uucp	Schedules BNU jobs for the **cron** daemon, including the **uudemon.hour** shell procedure.
/var/spool/uucp/*SystemName* **/***	Contain files waiting to be transferred.

Related Information

The **uucp** command, **uudemon.hour** command, **uustat** command, **uux** command.

The **cron** daemon, **uucico** daemon.

Understanding the BNU Daemons in *Networks and communication management*.

---

uusend Command

Purpose

Sends a file to a remote host.

Syntax

uusend [**-m***Mode*] [**-r**] *Sourcefile System* [**!***System* ...] **!** *RemoteFile*

Description

The **uusend** command sends a file to a given location on a remote system. The remote system need not be directly connected to the local system, but a chain of UUCP links must connect the two systems, and the **uusend** command must be available on each system in the chain.

The chain of systems is given by the *System*[**!***System* ...] parameter, which lists each remote system the file is to be transferred to, separated by ! (exclamation points). The **!***Remotefile* parameter gives the name under which the file is to be stored when it reaches the last system in the chain.

> **Note:** Do not put any spaces between the system names and exclamation points or between the last exclamation point and the remote file name.

The *SourceFile* parameter specifies the name of the file on the local system. If a **-** (dash) is used, the **uusend** command uses standard input.

Flags

-m *Mode*	Specifies that the mode of the file on the remote system will be taken from the octal number given. If this flag is not specified, the mode of the input file will be used.
-r	Prevents the starting of the **uucico** daemon, which transfers files between systems. The default is to start the **uucico** daemon.

The flags are primarily used internally by the **uusend** command when it is transferring files to the next remote system in the chain.

Example

To send a file across one system to another system, enter:

```
uusend /etc/motd nostromo!gandalf!~nuucp
```

The `/etc/motd` file is sent to system `nostromo` and then to system `gandalf`, and placed in `nuucp`'s home directory, `/var/spool/uucppublic/nuucp`, where `nuucp` is a BNU login ID.

Files

/usr/bin/uusend
Contains the **uusend** command.

Related Information

The **uucp** command, **uux** command.

The **uucico** daemon.

---

uusnap Command

Purpose

Displays the status of BNU contacts with remote systems.

Syntax

uusnap

Description

The **uusnap** command displays a table showing the status of the Basic Networking Utilities (BNU). The table includes the following information for each remote system:

`SystemName`	Specifies the name of the remote system.
`Number Cmds`	Specifies the number of command files (**C.*** files) queued for the remote system.
`Number Data`	Specifies the number of data transfers (**D.*** files) queued for the remote system.
`Number Xqts`	Specifies the number of remote command executions (**X.*** files) queued for the remote system.
`Message`	Specifies the current status message for the site, from the **/var/spool/uucp/.Status/** *SystemName* file. The `Message` field may include the time remaining before BNU can retry the remote system, and the count of the number of times (if any) BNU has tried unsuccessfully to reach the system.

Example

To see a snapshot of the status of BNU, enter:

```
uusnap
```

The output from this command is similar to the following:

```
nostromo  4 Cmds  2 Data  2 Xqts  SUCCESSFUL
zeus      2 Cmds  1 Data  2 Xqts  NO DEVICES AVAILABLE
```

These lines indicate that four command files, two data files, and two execute files are currently queued for system `nostromo`. The last connection to `nostromo` was successful. The last attempt to contact system `zeus`, on the other hand, was not successful because no device was available on the local system.

Files

/usr/bin/uusnap	Contains the **uusnap** command.
/var/spool/uucp/.Status/*SystemName*	Records the status of BNU contacts with a remote system.
/var/spool/uucp/*SystemName*	Contains **C.***, **D.***, and **X.*** files to be transferred by the **uucico** daemon.
/var/spool/uucp/*SystemName***/C.***	Instruct BNU about files to be transferred.
/var/spool/uucp/*SystemName***/D.***	Contain files to be transferred by BNU.
/var/spool/uucp/*SystemName***/X.***	Specify commands to be remotely executed by BNU.

Related Information

The **uucp** command, **uux** command, **uuq** command.

The **uucico** daemon.

BNU file and directory structure, BNU maintenance commands in *Networks and communication management*.

---

uustat Command

Purpose

Reports the status of and provides limited control over BNU operations.

Syntax

uustat [[**-n** *Number*] [**-a** | **-k** *JobID* | **-m** | **-p** | **-q** | **-r** *JobID*] | [**-s** *System*] [**-u** *User*]]

Description

The **uustat** command is a Basic Networking Utilities (BNU) command that displays status information about several types of BNU operations. It is particularly useful in monitoring the status of BNU requests.

In addition, the **uustat** command also gives a user limited control over BNU jobs queued to run on remote systems. By issuing the command with the appropriate flag, a user can check the general status of BNU connections to other systems and cancel copy requests made with the **uucp** and **uuto** commands.

If the **uustat** command is issued without any flags, the command reports the status of all BNU requests issued by the current user since the last time the holding queue was cleaned up. Such status reports are displayed in the following format:

```
jobid date/time status system_name user_ID size file
```

There are two types of BNU queues:

- The current queue, accessed with the **-q** flag, lists the BNU jobs either queued to run on or currently running on one or more specified computers.
- The holding queue, accessed with the **-a** flag, lists all jobs that have not executed during a set period of time.

After the time has elapsed, the entries in the holding queue are deleted either manually with the BNU **uucleanup** command or automatically by commands such as **uudemon.cleanu** started by the **cron** daemon.

When sending files to a system that has not been contacted recently, it is a good idea to use the **uustat** command to see when the last access occurred; the remote system may be down or out of service.

Flags

The following flags are mutually exclusive. Use only one at a time with the **uustat** command.

-a Displays information about all the jobs in the holding queue, regardless of the user who issued the original BNU command.

-k*JobID* Cancels the BNU process specified by the *JobID* variable. The person using this flag must either be the one who made the **uucp** request now being canceled or be operating with root authority.

This flag cancels a process only when that job is still on the local computer. After BNU has moved the job to a remote system for execution, the **-k** *JobID* flag cannot be used to cancel the remote job.

-m Reports the status of the most recent attempt to contact the specified system with a BNU command. If the BNU request was completed, the status report is successful. If the job was not completed, the status report is an error message saying that the login failed.

-n *Number* Allows the user to specify the amount of machines from which to collect BNU status information. The amount specified should be greater than or equal to the amount of machines in the Systems file. The default is 200.

-p Runs a **ps -flp** (process status: full, long list of specified process IDs) for all PID numbers in the lock files.

-q Lists the jobs currently queued to run on each system. These jobs are either waiting to execute or in the process of executing. If a status file exists for the system, its date, time, and status information are reported. When the job is finished, BNU removes that job listing from the current queue.

In a status report, a number in parentheses next to the number of a **C.*** (command) file or an **X.*** (execute) file represents the age in days of the oldest **C.*** or **X.*** file for that system. The **retry** field represents the number of times BNU tried and failed to execute the command because of, for example, a failed login, locked files, or an unavailable device.

-r*JobID* Marks the files in the holding queue specified by the *JobID* variable with the current date and time. Use this flag to ensure that a cleanup operation does not delete files until the job's modification time reaches the end of the specified period.

You can use either one or both of the following flags with the **uustat** command:

-s *System* Reports the status of BNU requests for the workstation specified by the *System* variable. The *System* name can contain only ASCII characters.

-u *User* Reports the status of BNU requests by the user specified by the *User* variable, for any workstation. The *User* name can contain only ASCII characters.

Examples

1. To display the status of all BNU jobs in the holding queue, type:

```
uustat  -a
```

The system responds with a message similar to the following:

```
heraC3113   11/06-17:47 S  hera    amy  289   D.venus471afd8
zeusN3130   11/06-09:14 R  zeus    geo  338   D.venus471bc0a
merlinC3120 11/05-16:02 S  merlin  amy  828   /home/amy/tt
merlinC3119 11/05-12:32 S  merlin  msg  rmail amy
```

Field	Description
1	Job ID of the operation
2	Date and time the BNU command was issued
3	An S or an R, depending on whether the job is to send or receive a file
4	Name of the system on which the command was entered
5	User ID of the person who issued the command
6	Size of the field or the name of the remote command
7	Name of the file.

When the size of the file is given, as in the first three lines of the example output, the file name is also displayed. The file name can be either the name given by the user, as in the `/home/amy/tt` entry, or a name that BNU assigns internally to data files associated with remote executions, such as `D.venus471afd8`.

2. To display the status of all jobs in the current queue, type:

```
uustat  -q
```

The system responds with a message similar to the following:

```
merlin  3C       07/15-11:02   NO DEVICES AVAILABLE
hera    2C       07/15-10:55   SUCCESSFUL
zeus    1C (2)   07/15-10:59   CAN'T ACCESS DEVICE
```

This output tells how many **C.*** (command) files are waiting for each system. The number in parentheses `(2)` in the third line of the example indicates that the **C.*** file has been in the queue for two days.The date and time refer to the current interaction with the system, followed by a report of the status of the interaction.

3. To display all process IDs in the lock file, type:

```
uustat  -p
```

The system responds with a message similar to the following:

```
LCK..tty0:  881
LCK.S.0:  879
LCK..hera:  881
F   S UID  PID  PPID  C  PRI NI ADDR SZ  WCHAN    STIME    TTY
101 S uucp 881  879   26 39  39 370  296 3fffe800 09:57:03  -
TIME  COMD
0:00  UUCICO  -r1  -shera
101 S uuc  879  1     11 33  39 770  156 8d874    09:57:02  -
0:00  /usr/sbin/uucp/uusched
```

4. To cancel a job in the current queue, first determine its job ID and then issue the command to cancel the job. To determine the job ID, type:

```
uustat  -a
```

The system responds with a message similar to the following:

```
heraC3113   11/06-17:47  S  hera    amy  289  D.venus471afd8
merlinC3119 11/06-17:49  S  merlin  geo  338  D.venus471bc0a
```

To cancel the job with the ID of `heraC3113`, type:

```
uustat  -k heraC3113
```

5. To report the status of jobs requested by system `hera`, type:

```
uustat  -s hera
```

The system responds with a message similar to the following:

```
heraN1bd7  07/15-12:09  S  hera  amy  522    /usr/amy/A
heraC1bd8  07/15-12:10  S  hera  amy   59    D.3b2a12ce4924
heraC3119  07/15-12:11  S  hera  amy  rmail   msg
```

Files

/etc/locks	Contains lock files to prevent multiple uses of devices.
/usr/bin/uustat	Specifies the command pathname.
/var/spool/uucp	Contains BNU status information.

Related Information

The **cron** daemon, **ct** command, **cu** command, **echo** command, **stty** command, **uucleanup** command, **uucp** command, **uuname** command, **uupick** command, **uuto** command, **uux** command.

---

uuto Command

Purpose

Copies files from one system to another.

Syntax

uuto [**-m**] [**-p**] *Source ... User*

Description

The **uuto** command is a Basic Networking Utilities (BNU) command that copies one or more *Source* files from one system to a specified *User* on another UNIX based system. This program uses the **uucp** command for the actual file transfer, but the **uuto** command enables the recipient to use the **uupick** command options to handle the transferred file on the local system.

The sender issues the **uuto** command to copy one or more files to a specific user ID on another system. The **uucp** command then copies the file to the BNU public directory, **/var/spool/uucppublic**, on the destination system. The **uucp** command also creates an additional subdirectory called **receive** (if it does not already exist) and directories below it in which to hold the files until the recipient retrieves them with the **uupick** command. The full path names to the copied files are some form of the following name:

/var/spool/uucppublic/receive/*UserName***/***System***/***File*

where the *UserName* and *System* directories are created based on the *User* parameter given with the **uuto** command.

Once the copied file is in the **receive** directory, the **rmail** command notifies the recipient that a file has arrived. The recipient then issues the **uupick** command, and this command searches the public directory for files sent to the recipient and notifies the recipient about each file it locates. The recipient then enters one of the **uupick** options to handle the file.

Source and Destination File Names

The sender must give the name of the file to be sent and user and system to which the file is to be transferred. The *Source* parameter is the path name of the source file. This can be the name of the file if the file is in the directory from which the **uuto** command is issued. If the file is in a different directory, the complete or relative path name of the file must be given.

The *User* parameter is the path name to the specific location where the source file is to be copied. This path name must include the user identification of the person the file is being sent to. The *User* parameter has the form:

System!*UserName*

where *System* is the name of the remote system connected to the local system, and *UserName* is the login name of the recipient of the transferred files on the specified system.

When copying a file from one user to another user on the local system, omit the *System* entry; the destination is the ID of the user to whom the file is being sent. System names can contain only ASCII characters.

Flags

-m Notifies the sender by the **bellmail** command when the source file has been successfully copied.

-p Copies the source file to the spool directory on the local system. The source file resides in the spooling directory for a set period of time (defined in the **uusched** program) before the **uucp** command calls the **uucico** daemon, which actually transfers the copy to the public directory on the specified remote system. The default is to transfer a source file directly to the specified user.

Examples

1. To copy a file to a user on a remote system, enter:

   ```
   uuto /home/bin/file1 zeus!karen
   ```

   In this example, the `/home/bin/file1` file is sent to user `karen` on the remote system `zeus`.
2. To copy a file to a user on a remote system and be notified whether the source file was successfully copied, enter:

   ```
   uuto  -m /home/bin/file2 zeus!karen
   ```

   In this example, the `/home/bin/file2` file is sent to user `karen` on the remote system `zeus` and a message is returned to the sender verifying that the copy was successful.
3. To copy a file to another user on your local system, enter:

   ```
   uuto /home/bin/file3 ron
   ```

   In this example, the `/home/bin/file3` file is sent to user `ron` on the local system. No mail message is sent to the recipient in a local transfer.

Files

/usr/bin/uuto	Contains the **uuto** command.
/var/spool/uucppublic	Is the BNU public directory.

Related Information

The **bellmail** command, **ct** command, **cu** command, **uucp** command, **uuname** command, **uupick** command **uustat** command, **uux** command.

The **uucico** daemon, **uusched** daemon.

uutry Command

Purpose

Contacts a specified remote system with debugging turned on and allows the user to override the default retry time.

Syntax

uutry [**-x***DebugLevel*] [**-r**] *SystemName*

Description

The **uutry** command contacts a remote system, specified by the *SystemName* parameter, using debugging mode. Debugging mode provides a means of monitoring Basic Networking Utilities (BNU) connections to remote computers and file transfers. The **uutry** command calls the **uucico** daemon to contact the remote system.

The debugging output is scrolled on the screen of the local system. Once the system has finished displaying this information, press the Interrupt key to return to the prompt.

The **-r** flag overrides the default retry time if the first attempt to contact the remote system is unsuccessful. The default retry time is 5 minutes.

The *SystemName* parameter, which is required, specifies the name of the remote system you wish to contact.

Notes:

1. Either you must be in the **/usr/sbin/uucp** directory when you issue the **uutry** command or you must issue the command with the full path name, **/usr/sbin/uucp/uutry.**
2. The **uutry** command is a shell script stored in the **/usr/lib/uucp** directory.
3. If the debugging output scrolls too quickly to be read, use the **Uutry** command to save the output in a temporary file.

Flags

-r Overrides the default retry time. If for some reason the **uucico** daemon cannot complete the requested connection, the daemon waits for a set amount of time and tries again. The default retry time is 5 minutes.

> **Note:** The time at which the remote system was last polled is recorded in the *SystemName* file in the **/var/spool/uucp/.Status** directory.

-x*DebugLevel* Overrides the default amount of detail in the debugging information that the **uutry** command displays on the screen. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the final report to be more detailed. If the **-x** flag is not used, the **uucico** daemon is started with the default level, which produces a moderate amount of information.

Examples

1. To change the amount of detail the **uutry** command provides about the progress of the **uucico** operation, use the **-x** flag to specify a different debugging level. For example, entering:

   ```
   /usr/sbin/uucp/uutry  -x9 venus
   ```

   instructs the **uutry** command to generate as much information as possible about the way in which the **uucico** daemon is working.

2. The default time at which to retry a contact to a remote system when the first contact was unsuccessful is 5 minutes. To shorten the default retry time for contacting the remote system, enter:

   ```
   /usr/sbin/uucp/uutry  -r venus
   ```

   Using the **-r** flag instructs the **uucico** daemon to contact remote system `venus`, overriding the default retry time. The daemon attempts to contact system `venus`, retrying periodically until the connection is successful, and then produces debugging output on the display screen of the local system.

Files

/usr/sbin/uucp/uutry	Contains the **uutry** command.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dial-code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/.Status/*SystemName*	Lists the last time the remote system named by the *SystemName* file was contacted.
/var/spool/uucppublic/*	Contain the BNU public directories.

Related Information

The **tail** command, **uucp** command, **Uutry** command, **uukick** command, **uux** command.

The **uucico** daemon, **uucpd** daemon.

Monitoring a BNU remote connection and Monitoring a BNU file transfer in *Networks and communication management*.

Maintaining BNU, BNU daemons in *Networks and communication management*.

Uutry Command

Purpose

Contacts a specified remote system with debugging turned on and saves the debugging output in a temporary file.

Syntax

Uutry [**-x***DebugLevel*] [**-r**] *SystemName*

Description

The **Uutry** command contacts a remote system, specified by the *SystemName* parameter, using debugging mode. Debugging mode provides a means of monitoring Basic Networking Utilities (BNU) connections to remote computers and file transfers.

The **Uutry** command starts the **uucico** daemon, which actually contacts the specified system. The **uucico** daemon produces debugging output that enables you to monitor the daemon's progress as it establishes the connection to the remote system, performs the remote login, and transfers a file.

The debugging output is scrolled on the screen of the local system. Once the system has finished displaying this information, press the Interrupt key to return to the prompt.

In addition to displaying the debugging output on the screen, the **Uutry** command directs this information to a file named **/tmp/***SystemName*, where the *SystemName* parameter is the name of the remote system you are attempting to contact. Again, when the last of the output has been displayed, press the Interrupt key to return to the prompt.

The *SystemName* parameter, which is required, specifies the name of the remote system you wish to contact.

Notes:

1. Press the Interrupt key while the system is scrolling the output generated by the **Uutry** command to return to the prompt. The **uucico** daemon continues to place the debugging information in the **/tmp/***SystemName* file.
2. Either you must be in the **/usr/sbin/uucp** directory when you issue the **Uutry** command or you must issue the command with the full path name, **/usr/sbin/uucp/Uutry**.
3. The **Uutry** command is a shell script stored in the **/usr/sbin/uucp** directory.

Flags

-r Overrides the default retry time. If for some reason the **uucico** daemon cannot complete the requested connection, the daemon waits for a set amount of time and tries again. The default retry time is 5 minutes.

> **Note:** The time the remote system was last polled is recorded in the **/var/spool/uucp/.Status/***SystemName* file.

-x*DebugLevel* Overrides the default amount of detail in the debugging information that the command displays on the screen. The valid range for the *DebugLevel* variable is 0 to 9, with a default of 5. Higher numbers cause the final report to be more detailed. If the **-x** flag is not used, the **uucico** daemon is started with the default level, which produces a moderate amount of information.

Example

To change the amount of detail the **Uutry** command provides about the progress of the **uucico** operation, use the **-x** flag to specify a different debugging level. For example, entering:

```
/usr/sbin/uucp/Uutry  -x9 venus
```

instructs the **Uutry** command to generate as much information as possible about the way in which the **uucico** daemon is working.

Files

/tmp/*SystemName*	Contains debugging output from the **Uutry** command (temporary file).
/usr/sbin/uucp/Uutry	Contains the **Uutry** command and all the configuration files for BNU.
/etc/uucp/Devices	Contains information about available devices.
/etc/uucp/Dialcodes	Contains dial-code abbreviations.
/etc/uucp/Dialers	Specifies initial handshaking on a connection.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/etc/uucp/Systems	Describes accessible remote systems.
/var/spool/uucp/.Status/*SystemName* file	Lists the last time a remote system was contacted.

/var/spool/uucppublic/*	Contain the BNU public directories.

Related Information

The **uucico** daemon, **uucpd** daemon.

The **tail** command, **uucp** command, **uustat** command, **uutry** command, **uukick** command, **uux** command.

How to Monitor a BNU Remote Connection and Monitoring a BNU remote connection in *Networks and communication management*.

Maintaining BNU, Understanding the BNU Daemons in *Networks and communication management*.

---

uux Command

Purpose

Runs a command on another UNIX-based system.

Syntax

uux [**-c** | **-C**] [**-n** | **-z**] [**-**] [**-a***Name*] [**-b**] [**-g***Grade*] [**-j**] [**-p**] [**-e**] [**-r**] [**-s***File*] [**-x***DebugLevel*] *CommandString*

Description

The **uux** command is a Basic Networking Utility (BNU) that runs a specified command on a specified UNIX-based system while enabling the user to continue working on the local system. Before running the requested command, the **uux** command gathers any necessary files from the designated systems. The user can direct the output from the command to a specific file on a specific system. For security reasons, many installations permit the **uux** command to run only the **rmail** command.

The **uux** commands on other systems create execute (**X.***) files that run commands on the local system. In addition, the **uux** command on the local system creates both command (**C.***) files and data (**D.***) files for transfer to other systems. Execute files contain the command string to be executed on the designated system. Command files contain the same information as those created by the **uucp** command. Data files either contain the data for a remote command execution or else become **X.*** files on remote systems for remote command executions.

The full path name of an execute file is a form of the following:

/var/spool/uucp/*System***/X.***SystemNxxxx*

After creating the files in the spooling directory, the **uux** command calls the **uucico** daemon to transfer the files from the spooling directory on the local system to the designated remote system. Once the files are transferred, the **uuxqt** daemon on the remote system executes the *CommandString* on the specified system, placing any output from the command in the file designated by the original **uux** command request.

The *CommandString* argument is made up of one or more arguments that look like an operating system command line, except that *CommandString* argument may be prefixed by the name of the remote system in the form *System***!**. The default *System* is the local system. Unless the user entering the **uux** command includes the **-n** flag, the command notifies that user if the remote system does not run the command. This response comes by mail from the remote system.

Source and Destination File Names

- When specifying the destination of the output of a command, the **uux** command can be entered in either one of the following formats:

  - **uux** [*Options*] ″*CommandString*> *Destination*″
  - **uux** [*Options*] *CommandString*\ {*Destination*\}.
- Destination names can be either of the following:
  - A full path name
  - A full path name preceded by ~*User*, where *User* is a login name on the specified system. The **uux** command replaces this path name with the user's login directory.
- The shell pattern-matching characters ? (question mark), * (asterisk), and [...] (brackets) can be used in the path name of a source file (such as files compared by the **diff** command); the appropriate system expands them. However, using the * character may occasionally produce unpredictable or unanticipated results. Shell pattern-matching characters should not be used in the destination path name.
- Place either two backslashes (\ . . . \) or a pair of quotation marks (″ . . . ″) around pattern-matching characters in a path name so the local shell cannot interpret them before the **uux** command sends the command to a designated system.
- If you are using the special shell characters > (greater than), < (less than), ; (semicolon), or | (vertical bar) in a path name, place either \ . . . \ or ″ . . . ″ around the individual character or around the entire command string.
- Do not use the shell redirection characters << or >> in a path name.
- The **uux** command attempts to move all files specified on the command line to the designated system. Enclose the names of all output files in parentheses so that the **uux** command does not try to transfer them.
- When specifying a *System*, always place it before the *CommandString* argument in the entry. System names can contain only ASCII characters.
- The ! (exclamation point) preceding the name of the local system in a command is optional. If you choose to include the ! to run a command on the local system using files from two different remote systems, use ! instead of *System*! to represent the local system, and add *System*! as the first entry in any path name on the remote systems.
- The exclamation point representing a system in BNU syntax has a different meaning in C shells. When running the **uux** command in a C shell, place a \ (backslash) before the exclamation point in a system name.

> **Note:** The notation ~ (tilde) is the shorthand way of specifying the public spooling directory, **/var/spool/uucppublic**.

Flags

- Makes the standard input to the **uux** command the standard input to the *CommandString* argument.

-a*Name* Replaces the user ID of the person issuing the command with the user ID specified with the *Name* variable.

-b Returns standard input to the command if the exit status is not zero.

-c Transfers the source files to the destination on the specified system. The source files are copied into the spooling directory, and the **uucico** daemon is invoked immediately. This flag is the default.

-C Transfers the source files to the spool directory. After a set period of time (specified in the **uusched** program), the **uucico** daemon attempts to transfer the files to the destination on the specified computer.

Occasionally, there are problems in transferring a source file; for example, the remote computer may not be working or the login attempt may fail. In such cases, the file remains in the spool directory until it is either transferred successfully or removed by the **uucleanup** command.

-e Enables file expansion.

-g*Grade*	Specifies when the files are to be transmitted during a particular connection. The *Grade* variable specifies a single number (0 through 9) or letter (A through Z, a through z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest). The default is **N**.
-j	Displays the job identification number of the process that is running the command on the specified system. Use this job ID with the BNU **uustat** command to check the status of the command or with the **uustat -k** flag to terminate the process.
-n	Prevents user notification by the **mail** command of the success or failure of a command. The default is to notify the user if the command fails.
-p	Uses the standard input to the **uux** command as the standard input to the *CommandString* argument. A **-** (minus) has the same effect.
-r	Prevents the starting of the spooling program that transfers files between systems. The default is to start the spooling program.
-s*File*	Reports the status of the transfer in a file specified by the *File* variable on the designated system. File names can contain only ASCII characters.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable must be a number from 0 to 9. A higher number gives a more detailed report.
-z	Notifies the user if the command completes successfully. This flag is the opposite of the system default, which is to notify the user only in the event of a failure.

Examples

1. To run the **qprt** command on a remote system, enter:

   ```
   uux merlin!qprt /reports/memos/lance
   ```

   In this example, the remote file `/reports/memos/lance` is printed on remote system `merlin`. Since neither the **-n** nor **-z** flag is specified, the **uux** command notifies the user only if the remote system fails to run the command. The response comes by the **mail** command from the remote system.

2. To run commands on two remote systems, enter the information on separate command lines:

   ```
   uux merlin!qprt /reports/memos/lance
   uux zeus!qprt /test/examples/examp1
   ```

   In this example, the remote `/reports/memos/lance` file is printed on remote system `merlin`, and the remote `/test/examples/examp1` file is printed on remote system `zeus`. Since neither the **-n** nor **-z** flag is specified, the **uux** command notifies the user only if the remote system fails to run the command. The response comes by the **mail** command from the remote system.

3. To queue a job that compares a file on the local system with a file on a remote system, using the **diff** command on the local system, and get the job ID of the job, enter:

   ```
   uux  -j "/usr/bin/diff /usr/amy/f1 hera!/home/amy/f2 > ~/f1.diff"
   ```

   In this example, the `/usr/amy/f1` file on the local system is compared to the `/home/amy/f2` file on the remote system `hera` and the output is placed in the `f1.diff` file in the local public directory (the full path name of this file is `/var/spool/uucppublic/f1.diff`). The destination name must be entered either preceded by a > with the whole command string enclosed in " " (quotation marks) or entered enclosed in braces and backslashes, as **\{** *DestinationName* **\}**. The **-j** flag causes the **uux** command to return the BNU job ID of the job.

4. To use the **diff** command on the local system to compare files that are located on two different remote systems, enter:

   ```
   uux "!/usr/bin/diff hera!/usr/amy/f1 venus!/home/amy/f2 > \ !f1.diff"
   ```

   In this example, the `/usr/amy/f1` file from the remote system `hera` is compared to the `/home/amy/f2` file from the remote system `venus` and the output is placed in the file `f1.diff`, located in the current working directory on the local system.

The output file must be write-enabled. If you are uncertain about the permission status of a specific target output file, direct the results to the public directory. The exclamation points representing the local system are optional. The destination name must be entered either preceded by a > with the whole command string enclosed in " " (quotation marks) or entered enclosed in braces and backslashes, as **\{** *DestinationName* **\}**.

5. To execute the **diff** command on two separate files from different systems, enter:

   ```
   uux "hera!/usr/bin/diff /tmp/out1 zeus/tmp/out2 > ~/DF"
   ```

   In this example, the `diff` file is on the remote system `hera`. The first source file is on the remote system `hera`, and the secondfile is on the system `zeus`. (`zeus` may be the local system or another remote system.) The output is directed to the file DF in the public directory on the local system.

6. To specify an output file on a different remote system, enter:

   ```
   uux hera!uucp venus!/home/amy/f1 \{merlin!/home/geo/test\}
   ```

   In this example, the **uucp** command is run on the remote system `hera`, and the `/home/amy/f1` file, stored on system `venus`, is sent to user `geo` on system `merlin` as `test`. The destination name is entered enclosed in braces and backslashes.

7. To get selected fields from a file on a remote system and place them in a file on the local system, enter:

   ```
   uux "cut -f1 -d: hera\!/etc/passwd > ~/passw.cut"
   ```

   In this example, the **cut** command is run on the local system. The first field from each line of the password file on system `hera` is placed in the `passw.cut` file in the public directory on the local system. The **uux** command is running in a C shell, so a \ (backslash) must precede the exclamation point in the name of the remote system.

8. To use the **uux** piping option to specify a remote copy of the `/tmp/example` file to `/tmp/examplecopy` on system `mercury` use the following syntax:

   ```
   uux -p mercury!
   cp /tmp/example /tmp/examplecopy
   ```

   The user must enter a Ctrl-D in order to terminate the command input. After Ctrl-D is pressed, the command will be spooled for remote execution on system `mercury`.

Files

/usr/bin/uux	Contains the **uux** command.
/var/spool/uucp	Is the spooling directory.
/var/spool/uucppublic	Is the public directory.

Related Information

The **ct** command, **cu** command, **mail** command, **rmail** command, **uucleanup** command, **uucp** command, **uuname** command, **uupick** command, **uustat** command, **uuto** command, **qprt** command.

The **sendmail** daemon, **uucico** daemon, **uuxqt** daemon.

---

uuxqt Daemon

Purpose

Executes Basic Networking Utilities (BNU) remote command requests.

Syntax

uuxqt [**-e**] [**-s***SystemName*] [**-x***DebugLevel*]

Description

The Basic Networking Utilities (BNU) **uuxqt** daemon executes commands on designated remote systems.

The **uuxqt** daemon on each networked system periodically searches the spool directory for remote execute (**X.***) files. These files are sent to the directory by the **uucico** daemon in response to a **uux** command.

When it finds **X.*** files, the **uuxqt** daemon checks each file to make sure that:

- All the required data (**D.***) files are available.
- The requesting system has the necessary permissions to access the data files and run the requested commands.

> **Note:** The **uuxqt** daemon uses the **/etc/uucp/Permissions** file to validate file accessibility and command execution permission.

If the data files are present and the requesting system has the appropriate permissions, the **uuxqt** daemon executes the commands.

> **Note:** The **uuxqt** command is usually executed from the **uudemon.hour** command, a shell procedure, and not entered from the command line. You must have root user privileges to issue the **uuxqt** command from the command line.

Flags

-e	Enables file expansion.
-s*SystemName*	Designates the remote system to be contacted. Use only when starting the **uuxqt** command manually. The system name is supplied internally when the **uuxqt** command is started automatically. **Note:** System names can contain only ASCII characters.
-x*DebugLevel*	Displays debugging information on the screen of the local system. The *DebugLevel* variable is a single digit between 0 and 9, with a default of 5. The higher the *DebugLevel* variable, the more detailed the debugging information.

Security

Access Control: You must have root authority to start the **uuxqt** daemon from the command line.

Example

To start the **uuxqt** daemon for debugging, enter:

```
/usr/sbin/uucp/uuxqt  -svenus  -x7
```

This instructs the command to contact remote system `venus` and provide fairly detailed information about the contact.

Files

/usr/sbin/uucp/uuxqt	Contains the **uuxqt** daemon.
/etc/locks	Contains lock files that prevent multiple uses of devices and multiple calls to systems.

/etc/uucp/Maxuuxqts	Limits remote command executions.
/etc/uucp/Permissions	Describes access permissions for remote systems.
/var/spool/uucp/*	Contain the execute and data files.

Related Information

The **uucp** command, **uudemon.hour** command, **uustat** command, **uux** command.

The **cron** daemon, **uucico** daemon.

Understanding the BNU File and Directory Structure, Understanding the BNU Daemons, Understanding BNU Security in *Networks and communication management*.

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 003
11400 Burnet Road
Austin, TX 78758-3498
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

© Copyright IBM Corp. 1997, 2006

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX
AIX 5L
DPI
HACMP
IBM
Micro Channel
PowerPC
System/370

Java and all Java-based trademarks and logos are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be the trademarks or service marks of others.

Index

Special characters

/etc/utmp
  monitor 596
/etc/uucp/Permissions
  checking 597

Numerics

128-port asynchronous controller
  querying characteristics 235
  setting characteristics 235

A

accounting system
  starting up
    using startup command 190
  summarizing records
    using sa command 1
  turning off
    using shutacct command 89
    using turnacct command 532
  turning on
    using turnacct command 532
acct/* commands
  shutacct 89
  startup 190
  turnacct 532
administration program
  for SCCS commands
    using sccs command 20
aliases
  removing 558
analyzing virtual memory snapshot
  memory management
    using svmon command 255
archive
  using tar command 315
arithmetic
  converting units 570
attaching serial lines
  TCP/IP
    using slattach command 93
auditing
  file installation in a secure system
    using sysck command 325

B

bibliographic database
  sorting
    using sortbib command 153
binary data
  storing in a file
    using sa1 command 3
binary file
  decoding for mail transmission
    using uuencode command 611, 617
  encoding for mail transmission
    using uuencode command 611, 617
  finding the printable strings
    using strings command 215
block count
  displaying a file's
    using sum command 241
BNU
  checking status of operations
    using uustat command 632
  cleaning of log files
    using uudemon.cleanu command 613
  cleaning of spooling directories
    using uudemon.cleanu command 613
  commands
    executing remotely 643
    running remotely 640
  communication between TCP/IP 610
  configuration information, entering
    using uucpadm command 608
  copying files between operating systems 604
  debugging
    saving output to a file 638
  debugging mode
    using 620
  debugging remote connections 599
  files
    completing transfer of 624
    converting to ASCII 617
    copying between systems 635
    scheduling transfers 629
    transferring between systems 599
  initiating transport calls
    using uudemon.hour command 615
  log files
    cleaning 621
    displaying 621
  networked computers
    listing 623
  polling remote systems 626
    using uudemon.poll command 616
  remote systems
    contacting in debug mode 638
  required files
    checking for 597
  spooling directories
    cleaning 601, 602
    deleting files from 602
    removing files from 601
  status
    obtaining 631
  status of file transfers
    uudemon.admin command 612
  tip command 381
    escape signals 382

© Copyright IBM Corp. 1997, 2006

BNU *(continued)*
tip command *(continued)*
variables 384
uucheck command 597
uucico daemon 599
uuclean command 601
uucleanup command 602
uucp 604
uucpadm command 608
uucpd daemon 610
uudemon.admin command 612
uudemon.cleanu command 613
uudemon.hour command 615
uudemon.poll command 616
uukick command 620
uulog command 621
uupoll command 626
uuq command 627
uusched daemon 629
uusend command 630
uusnap command 631
Uutry command 638
uuxqt daemon 643
BNU job queue
deleting entries
using uuq command 627

C

CA certificate
importing
using smimpcacert command 105
CA certificate exporting
using smexpcacert command 103
ca certificate listing 110
certificate authority
read-only information 100
certificate authority's certificate
importing
using smimpcacert command 105
certificate requests
generating
using smgenkeycr command 104
processing and generating 115
character translation 438
characters
translating
using tr command 423
checking
file installation in a secure system
using tcbck command 325
checksum
displaying a file's
using sum command 241
cleaning up
the sendmail queue
using smdemon.cleanu command 102
code set maps
setting 76
code sets 76
command
printing the time of execution
using time command 374
reporting the elapsed time for
using timex command 379
reporting the user time for
using timex command 379
smimpcacert 105
commands
sccs 20
sccshelp 25
slattach 93
sliplogin 96
smdemon.cleanu 102
smexpcacert 103
smgenkeycr 104
smgenprivkr 104
snapshot 124
snmpevent 127
startcondresp 179
startrpdomain 184
startrpnode 187
stopcondresp 197
stoprpdomain 200
stoprpnode 202
stty 228
su 238
sum 241
svmon 255
tbl 321
tcbck 325
timedc 378
tip 381
touch 410
tr 423
tracesoff 435
trcevgrp 442
troff 452
tset 517
tsh 519
tsm 521
tunchange 526
tuncheck 527
tvi 534
type 539
unfencevsd 564
updatevsdnode 578
updatevsdtab 580
updatevsdvg 581
usrck 590
uucpadm 608
uudemon.admin 612
uudemon.cleanu 613
uudemon.hour 615
uudemon.poll 616
uuq 627
uustat 632
conditional expressions
evaluating 363

connect to a remote system
BNU
using tip command 381
control scripts
topsvcsctrl 406
control, limited
of BNU operations
using uustat command 632
conversing with other users
using talk command 311
converting standard-input terminal line
to Serial Line Internet Protocol (SLIP) link
using sliplogin command 96
core dump size limits 550
core file
gathering core file 122
correctness, verifying
of user definition
using usrck command 590
counters
processing data from the operating system
using sadc command 5
sampling data from the operating system
using sadc command 5
saving data from the operating system
using sadc command 5
CPU
reporting usage 412

D

daemon
utmpd 596
daemons
tftpd 371
daily report
writing in a file
using sa2 command 3
data area size limits 550
deleting entries
BNU job queue
using uuq command 627
description of command type and arguments
using type command 539
device configuration commands
savebase 13
devices
customized
saving information about 13
Devices file format
setting up
using uucpadm command 608
Dialcodes file format
setting up
using uucpadm command 608
directory
unmounting
using umount command 555
dump device
changing the primary 294
changing the secondary 294
dump device *(continued)*
starting a kernel dump to the primary 298
starting a kernel dump to the secondary 298

E

edit status
displaying 4
editor
stream
using sed command 55
editors
tvi editor, calls
using tvi command 534
ERRM commands
snmpevent 127
ERRM scripts
snmpevent 127
errors
fixing in file
using tcbck command 328
escape signals
using tip command 382
exit values
returning 512
expressions
evaluating conditional 363

F

file
deleting repeated lines in a
using uniq command 568
fixing errors in
using tcbck command 328
splitting into pieces
using split command 159
unmounting
using umount command 555
file inclusion
processing
using soelim command 146
file mode creation masks 552
file size limits 550
file system
unmounting
using umount command 555
file systems
removing unwanted files
using skulker command 93
file transfers, BNU
status
uudemon.admin command 612
files
access times
updating 410
comparing two
using sdiff command 46
compression 562
copying between systems 604
decompression 561

files *(continued)*
displaying block count
using sum command 241
displaying comparison side-by-side of two
using sdiff command 46
displaying the checksum
using sum command 241
expanding
using unpack command 576
merging
using sort command 147
modification times
updating 410
removing ifdef'ed lines 567
SCCS
canceling specified versions 566
comparing two versions 24
displaying edit status 4
sorting
using sort command 147
sorting unordered lists 522
transferring with tftp command 365
writing
from specified point 309
folder
displaying messages in a
using scan command 18
formatting txt for
printing on typesetting devices
using troff command 452
FORTRAN
translating programs to RATFOR 226

G

games
tic-tac-toe 523
games directory permissions 533
groups
resetting for the current login session
using setgroups command 74

H

hlptcpdump 331
hlpuil 547
hosts
connecting local with remote
using telnet command 348
using tn command 348
using tn3270 command 348

I

i-node table
updating
using sync command 286
iconv library
generating conversion table for 543
ID, user
associated with session
using su command 238
importing
certificate authority's certificate
using smimpcacert command 105
inetd daemon
uucpd daemon and 610
init command 345
initiating transport calls
using BNU program
using uudemon.hour command 615
installing
files in a secure system
verifying using sysck command 325
internal certificate authority
defining 101
unconfiguring 117
Internet
tracing network packets 433
ip security crypto module 574

K

kernel messages
writing to terminal 583
kernel name list
generating a 444
key ring files, generating
server private
using smgenprivkr command 104

L

lines
deleting repeating
using uniq command 568
log files (BNU)
cleaning up 621
log files, BNU
cleaning
using uudemon.cleanu command 613
log, trace
formatting a report from
using trcrpt command 445
logical volume
copying one volume to a new volume 286
removing mirrors
using unmirrorvg command 574
split and copy 170
synchronizing mirrors that are not current
using syncvg command 287

M

mail bug report
mailing of 66
Mail commands
sendbug 66
sendmail 66
smdemon.cleanu 102

management information base variables
  managing with snmpinfo command 136
memory management
  analyzing virtual memory snapshot
    using svmon command 255
  updating the super block 578
message routing 174
messages
  listing lines of 18
  logs system 301
  sending
    using send command 63
  showing
    using show command 85
  sorting
    using sortm command 154
messages, SCCS
  displaying help information
    using sccshelp command 25
MH
  slocal command 100
  spost command 174

N

NCS commands
  uuid_gen 618
networked computers
  displaying list of 623
NFS commands
  showmount 87
  spray 175
NFS daemons
  sprayd 177
  statd 195
nroff command
  formatting table for
    using tbl command 321

O

object file
  finding the printable strings
    using strings command 215
object files
  displaying section sizes of XCOFF 92
  displaying symbol information with stripnm command 218
  reducing size of XCOFF 216

P

paging 282
  specifying additional devices for
    using swapon command 282
Permissions file format
  setting up
    using uucpadm command 608
  verifying 597
phones file format
  setting the phones variable 385
physical memory size limits 550
Poll file format
  setting up
    using uucpadm command 608
polling
  remote systems
    using uudemon.poll command 616
printer
  changing driver settings
    using splp command 172
  displaying driver settings
    using splp command 172
printing on typesetting devices
  formats text for
    using troff command 452
private key ring installation 107
process
  initializing
    using init command 345
    using telinit command 345
process resource allocation
  removing unused modules 95
process suspension
  suspending execution for an interval 95
processing incoming mail, MH 100
program
  copying output into a file 343
program loops
  returning exit values 512
program, administration
  for SCCS commands
    using sccs command 20

Q

querying characteristics
  terminals
    using stty command 228

R

Reliable Scalable Cluster Technology (RSCT) topology services
  control scripts
    topsvcsctrl 406
  scripts
    topsvcs 405
remote command requests
  executing 643
remote file format
  setting the remote variable
    using tip command 386
remote system
  connecting to
    using tip command 381
remote systems
  executing commands on 643
  polling
    using uudemon.poll command 616
resource limits 550
route mail for local or network delivery 66

RSCT topology services
control scripts
topsvcsctrl 406
scripts
topsvcs 405

S

sa command 1
sa1 command 3
sa2 command 3
sadc command 5
sar command 6
savebase command 13
savecore command 15
savevg command 16
scan command 18
SCCS
commands
administrating 20
files
administrating 20
canceling specified versions 566
comparing two versions 24
displaying edit status 4
help information 25
sccs command 20
SCCS commands
administration program for
using sccs command 20
displaying help information
using sccshelp command 25
sact 4
sccs 20
sccsdiff 24
sccshelp 25
unget 566
SCCS messages
displaying help information
using sccshelp command 25
sccshelp command 25
schedo command 25
scls command 43
screen
copying display to a file 343
creating a typescript 44
scripts
snap 120
snmpevent 127
topsvcs 405
topsvcsctrl 406
sctpctrl command 45
sdiff command 46
secldapclntd 49
secldifconv command 51
sectoldif command 52
securetcpip command 54
security
auditing the state of the system
using sysck command 325
sed command 55
sedmgr command 59
sendbug command 66
sendmail command 66
sendmail queue
cleaning up the
using smdemon.cleanu command 102
server
private key ring files, generating
using smgenprivkr command 104
server certificate importing 106
server private keys
generating
using smgenkeycr command 104
server properties listing 113
setclock command 72
setea command 73
setgroups command 74
setmaps command 76
setsenv command 79
settime command 80
setting characteristics
terminals
using stty command 228
setuname command 81
sh command 82
shell
executing with log in credentials
using the shell command 84
shell command 84
shell scripts
program loops
returning exit values 512
shells
default 82
show command 85
showmount command 87
shutacct command 89
shutdown command 89
size command 92
skulker command 93
slattach command 93
SLIP network interface
converting standard-input terminal line to
using sliplogin command 96
slattach command 93
sliplogin Command 96
slocal command 100
smcaprop command 100
smdefca command 101
smdemon.cleanu command
Mail 102
smexpcacert command 103
smgenkeycr command 104
smgenprivkr command 104
smimpcacert command 105
smimpservercert command 106
sminstkey command 107
smit command 108
smit.log file
redirecting 108, 110

smit.script file
redirecting 108, 110
smitty command 110
smlistcerts command 110
smrsh command 113
smserverprop command 113
smsigncert command 115
smtctl command 115
smundefca command 117
snap command 118
snap scripts 120
snapshot command 124
snapsplit command 125
SNMP
switching versions of snmpd agent daemon 142
SNMP version 1
Agent Applications
snmpdv1 command 130
start SNMP version 1 agent as background process 130
SNMP version 3
Agent Applications
snmpdv3 command 133
start SNMP version 3 agent daemon as background process 133
snmpd daemon 130
snmpdv1 daemon 130
snmpdv3 daemon 133
snmpevent command 127
snmpevent script 127
snmpinfo command 136
snmpmibd daemon 139
snmptrap command 141
snmpv3_ssw command 142
SNOBOL
compiling and interpreting 143
sodebug command 144
soelim command 146
sort command 147
sortbib command 153
sortm command 154
spaces
changing from tabs
using untab command 577
changing into tabs
using tab command 304
spell command 156
maintain hash lists for 156
spellin command 158
spelling list
creating
example of 158
using spellin command 158
verifying the absence of a word on
example of 159
using spellout command 158
spellout command 158
splat 160
split command 159
splitlvcopy command 170
splitvg command 168
splp command 172
spooling directories 601
spooling directories, BNU
cleaning
using uudemon.cleanu command 613
spost command 174
spray command 175
sprayd daemon 177
srcmstr daemon 178
standard input
copying to a file 343
creating typescript 44
start-secldapclntd 182
startcondresp command 179
startrpdomain command 184
startrpnode command 187
startsrc command 189
startup command 190
startx Command 192
statd daemon 195
status, reporting
of BNU operations
using uustat command 632
stop-secldapclntd 183
stopcondresp command 197
stoprpdomain command 200
stoprpnode command 202
stopsrc command 204
stpinet method 207
strace command 208
strchg command 209
strclean command 211
strconf command 211
STREAMS
displaying information 214
tunable parameters 224
STREAMS command
strchg 209
strconf 211
strload 220
STREAMS commands
scls 43
strace 208
strclean 211
STREAMS facility
configuration
changing 209
querying 211
driver names
listing 43
error log
receiving messages 212
error logger
cleaning up 211
modules
listing 43
portable environment
loading and configuring 220
strerr daemon 212
trace messages
printing 208

strerr daemon 212
strinfo command 214
strings command 215
stripnm command 218
strload command 220
strreset command 223
strtune command 224
sttinet method 227
stty command 228
stty-cxma command 235
style command 237
su command 238
subj command 240
subject list
  generating
    using subj command 240
subroutine call interface program 289
subroutine calls
  performing 289
subserver
  starting
    using startsrc command 189
  stopping
    using stopsrc command 204
  turning off tracing
    using tracesoff command 435
  turning on tracing
    using traceson command 437
subsystem
  starting
    using startsrc command 189
  stopping
    using stopsrc command 204
  turning off tracing
    using tracesoff command 435
  turning on tracing
    using traceson command 437
sum command 241
suma command 242
super block
  updating 578
survd daemon 252
svmon command 255
  command report 259
  detailed report 267
  frame report 267
  global report 257
  process report 261
  segment report 262
  user report 258
  workload management reports 260
    class report 260
    tier report 260
swap
  specifying additional devices for 282
swap command 280
swapon command 282
swapping 282
swcons command 284
swts command 285
sync command 286
synclvodm command 286
syncvg command 287
sysck command 290
syscorepath command 293
sysdumpdev command 294
sysdumpstart command 298
sysline command 300
syslogd daemon 301
system
  displaying uptime for the
    using uptime command 584
  ending operation of the
    using shutdown command 89
system activity
  collecting
    using sar command 6
  reporting
    using sar command 6
  saving
    using sar command 6
system call interface program 289
system calls
  performing 289
system console
  redirecting temporarily to a device
    using swcons command 284
  redirecting temporarily to a file
    using swcons command 284
system dump
  saving 15
system management
  performing
    using smit command 108
    using smitty command 110
System Resource Controller
  starting daemon
    using srcmstr daemon 178
system status
  displaying on terminal status line 300
Systems file format
  setting up
    using uucpadm command 608

T

tab command 304
tables
  formatting for nroff command
    using tbl command 321
  formatting for troff command
    using tbl command 321
tabs
  changing from spaces
    using tab command 304
  changing into spaces
    using untab command 577
talk command 311
talkd daemon 312
tape device
  consistency checking
    tapechk command 314

tape device *(continued)*
copying
tcopy command 330
giving subcommands to a streaming
using tctl command 341
tar Command 315
tbl command 321
tc command 324
tcbck command 325
modes of operation
check mode 326
update mode 327
security 328
TCP/IP
attaching serial lines
using slattach command 93
converting to SLIP
using sliplogin command 96
hosts
setting time and date 72
inet instance
disabling 207
enabling 227
internet instance
undefining 544
unloading 540
methods
udefinet 544
security feature
enabling 54
server function
support for talk command 312
support for TELNET protocol 359
server function for TFTP
using tftpd daemon 371
TCP sockets
tracing 508
time server daemon
invoking 375
timed daemon information
using timedc command 378
tracing Internet packets 433
tracking packets 508
TCP/IP commands
securetcpip 54
setclock 72
slattach 93
sliplogin 96
tftp 365
timedc 378
traceroute 433
trpt 508
utftp 365
TCP/IP daemons
talkd 312
telnetd 359
tftpd 371
timed 375
TCP/IP methods
stpinet 207
sttinet 227
TCP/IP methods *(continued)*
ucfgif 540
ucfginet 540
udefinet 544
tcpdump command 331
tee command 343
telinit command 345
telnet command 348
TELNET protocol
implementing
using telnet command 348
using tn command 348
using tn3270 command 348
telnetd daemon 359
telnet options 360
termdef command 362
terminal maps
setting 76
terminal sessions
making a typescript 44
terminal state manager
invoking
using tsm command 521
terminals
initializing
using tset command 517
manipulating kernel messages 583
querying characteristics
using stty command 228
using termdef command 362
setting characteristics
using stty command 228
using tset command 517
setting tab stops 305
specifying baud rate
using tset command 517
writing path names to standard output 524
terminfo descriptor files
translating from source to compiled format 373
test command 363
tftp command 365
tftpd daemon 371
tic command 373
time command 374
time stamps
updating
access times 410
modification times 410
timed daemon 375
manipulating with SRC 376
obtaining information about
using timedc command 378
timedc command 378
variables 378
timex command 379
tip command 381
escape signals 382
phones file format
setting the phones variable 385
remote file format
setting the remote variable 386

tip command *(continued)*
variables 384
tn command 348
tn3270 command 348
token-ring device driver
displaying statistics 387
tokstat command 387
topasout command 404
topology services subsystem
control scripts
topsvcsctrl 406
scripts
topsvcs 405
topsvcs script 405
topsvcsctrl script 406
touch command 410
tprof command 412
tput command 422
tr command 423
trace buffer
extracting from system dump image 441
trace event groups
add
using trcevgrp command 442
change
using trcevgrp command 442
delete
using trcevgrp command 442
trace log
formatting a report from
using trcrpt command 445
trace report
adding format templates
using trcupdate command 451
deleting format templates
using trcupdate command 451
replacing format templates
using trcupdate command 451
trace session
ending
using trcstop command 450
traceroute command 433
tracesoff command 435
traceson command 437
tracing, turning off
subservers or subsystems
using tracesoff command 435
translating
characters
using tr command 423
trbsd command 438
trcctl 440
trcdead command 441
trcevgrp command 442
trcnm command 444
trcrpt command 445
trcstop command 450
trcupdate command 451
troff command 452
command output interpreter for
using tc command 324
troff command *(continued)*
formatting table for
using tbl command 321
trpt command 508
output fields 508
trusted editors
provides
using tvi command 534
trusted shell
interpreting commands in a
using tsh command 519
invoking 520
Korn shell
differences between 520
tset command 517
tsh command 519
tsm command 521
tunchange command 526
tuncheck command 527
tundefault command 528
tunrestore command 529
tunsave command 531
turnacct command 532
tvi command 534
tvi editor
calling
using tvi command 534
customizing 535
limitations of 534
operating modes of 535
twconvdict command 537
twconvfont command 538
type command 539
type, command
writing description of
using type command 539
typesetting devices, printing on
formats text for
using troff command 452

U

ucfgif method 540
ucfginet method 540
uconvdef command 543
udefif method 544
udefinet method 544
udfcheck command 545
udfcreate command 546
udflabel command 546
uil command 547
UIL compiler
starting
using uil command 547
uimx command 548
ul command 549
ulimit command 550
umask command 552
umcode_latest command 553
umountall command 556
unalias command 558

uncompress command 561
underline
  performing
    using ul command 549
unexpand command 563
unfencevsd command 564
uniq command 568
unlink command 573
unlink subroutine 573
unloadipsec command 574
unmirrorvg command 574
unmount command 555
unpack command 576
untab command 577
updatevsdnode command 578
updatevsdtab command 580
updatevsdvg command 581
updating
  files in a secure system
    verifying using sysck command 325
uptime command 584
user
  changing session ID
    using su command 238
  re-initializing login session
    using shell command 84
  resetting protected state environment
    using setsenv command 79
user definition
  verifying correctness
    using usrck command 590
useradd command 584
userdel command 587
usermod command 588
users
  displaying compact list 590
usrck command 590
utftp command 365
utmpd 596
uucheck command 597
uucico command 599
uucico daemon 599
uuclean command 601
uucleanup command 602
uucp command 604
uucpadm command 608
uucpd command 610
uucpd daemon 610
uudecode command 611
uudemon.admin command 612
uudemon.admin shell script 612
uudemon.cleanu command 613
uudemon.cleanu shell script 613
uudemon.hour command 615
uudemon.hour shell script 615
uudemon.poll command 616
uudemon.poll shell script 616
uuencode command 617
uuid_gen command 618
UUIDs
  creating 618
uukick command 620
uulog command 621
uuname command 623
uupick command 624
uupoll command 626
uuq command 627
uusched command 629
uusched daemon 629
uusend command 630
uusnap command 631
uustat command 632
uuto command 624, 635
Uutry command 638
uux command 640
uuxqt command 643
uuxqt daemon 643

V

variables
  tip command 384
    setting 384
verifying
  correctness of user definition
    using usrck command 590
  file installation in a secure system
    using sysck command 325

W

writing
  with tabs restored 563
writing style
  analyzing
    using style command 237

X

X session
  initializing
    using startx command 192

Readers' Comments — We'd Like to Hear from You

AIX 5L™ Version 5.3
Commands Reference, Volume 5, s - u

Publication No. SC23-4892-03

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Name

Address

Company or Organization

Phone No.

Readers' Comments — We'd Like to Hear from You
SC23-4892-03

IBM

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Information Development
Department 04XA-905-6C006
11501 Burnet Road
Austin, TX 78758-3493

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Printed in U.S.A.

SC23-4892-03