Bull

AIX 5L Commands Reference

Volume 3/6, i – m

AIX

ORDER REFERENCE
86 A2 42EF 02

Bull

AIX 5L Commands Reference

Volume 3/6, i – m

AIX

Software

May 2003

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

ORDER REFERENCE
86 A2 42EF 02

The following copyright notice protects this book under the Copyright laws of the United States of America and other countries which prohibit such actions as, but not limited to, copying, distributing, modifying, and making derivative works.

Copyright © Bull S.A. 1992, 2003

Printed in France

Suggestions and criticisms concerning the form, content, and presentation of this book are invited. A form is provided at the end of this book for this purpose.

To order additional copies of this book or other Bull Technical Publications, you are invited to use the Ordering Form also provided at the end of this book.

Trademarks and Acknowledgements

We acknowledge the right of proprietors of trademarks mentioned in this book.

AIX® is a registered trademark of International Business Machines Corporation, and is being used under licence.

UNIX is a registered trademark in the United States of America and other countries licensed exclusively through the Open Group.

Linux is a registered trademark of Linus Torvalds.

The information in this document is subject to change without notice. Groupe Bull will not be liable for errors contained herein, or for incidental or consequential damages in connection with the use of this material.

Contents

About This Book ix
Who Should Use This Book ix
How to Use This Book ix
Highlighting ix
ISO 9000 xii
32-Bit and 64-Bit Support for the UNIX98 Specification xii
Related Information xii

Alphabetical Listing of Commands 1
ibm3812 Command 1
ibm3816 Command 2
ibm5585H-T Command 3
ibm5587G Command 4
iconv Command 5
id Command 6
ifconfig Command 8
ike Command 16
ikedb Command 21
imake Command 23
imapd Daemon 25
impfilt Command 26
importvg Command 27
imptun Command 29
inc Command 30
indent Command 32
indxbib Command 36
inetd Daemon 37
infocmp Command 39
infocenter Command 42
install Command 43
install_all_updates Command 45
install_assist Command 47
install_mh Command 48
install_wizard Command 48
installbsd Command 49
installp Command 50
installms Command 60
instfix Command 63
inucp Command 65
inudocm Command 66
inulag Command 67
inurecv Command 69
inurest Command 70
inurid Command 72
inusave Command 73
inutoc Command 76
inuumsg Command 76
invscout Command 77
invscoutd Command 80
ioo Command 86
iostat Command 96
ipcrm Command 100
ipcs Command 101

© Copyright IBM Corp. 1997, 2003

ipfilter Command . . . 104
ipreport Command . . . 105
ipsec_convert Command . . . 105
ipsecstat Command . . . 106
ipsectrcbuf Command . . . 106
iptrace Daemon . . . 107
isC2host Command . . . 109
isCChost Command . . . 110
istat Command . . . 111
jobs Command . . . 112
join Command . . . 114
joinvg Command . . . 118
kdb Command . . . 118
kdestroy Command . . . 124
keyadd Command . . . 126
keycomp Command . . . 127
keydelete Command . . . 129
keyenvoy Command . . . 130
keylist Command . . . 131
keylogin Command . . . 132
keylogout Command . . . 133
keypasswd Command . . . 134
keyserv Daemon . . . 135
kill Command . . . 136
killall Command . . . 138
kinit Command . . . 139
klist Command . . . 141
kmodctrl Command . . . 142
kpasswd Command . . . 143
krlogind Daemon . . . 144
krshd Daemon . . . 145
ksh Command . . . 147
kvno Command . . . 150
last Command . . . 151
lastcomm Command . . . 152
lastlogin Command . . . 154
lb_admin Command . . . 154
lb_find Command . . . 157
lbxproxy Command . . . 158
ld Command . . . 160
ldd Command . . . 178
ldedit Command . . . 179
learn Command . . . 180
leave Command . . . 181
lecstat Command . . . 182
lex Command . . . 184
libxrx Command . . . 190
line Command . . . 191
link Command . . . 192
lint Command . . . 193
listdgrp Command . . . 198
listvgbackup Command . . . 199
listX11input Command . . . 201
llbd Daemon . . . 201
ln Command . . . 202
locale Command . . . 204

localedef Command 206
lock Command 208
lockd Daemon 209
locktrace Command 211
elogevent, logevent Command 212
logform Command. 214
logger Command 215
login Command. 216
logins Command 218
logname Command 221
logout Command 222
look Command 222
lookbib Command 223
lorder Command 224
lp Command 225
lp.cat, lp.set, lp.tell Command 234
lpadmin Command 236
lpc Command 244
lpd Command 245
lpfilter Command 247
lpforms Command. 250
lpmove Command. 255
lppchk Command 256
lppmgr Command 258
lpq Command 260
lpr Command 263
lprm Command 267
lpsched Command 270
lpstat Command 271
lpsystem Command 275
lptest Command 277
lpusers Command. 277
ls Command 278
ls-secldapclntd Command 282
lsactdef Command 283
lsallq Command 287
lsallqdev Command 288
lsattr Command 289
lsaudrec Command 294
lsauthent Command 298
lsC2admin Command 298
lsCCadmin Command 299
lscfg Command. 300
lsclass Command 302
lscomg Command. 303
lscondition Command 306
lscondresp Command 310
lsconn Command 314
lscons Command 316
lsdev Command 317
lsdisp Command 324
lsfilt Command 325
lsfont Command 326
lsfs Command 327
lsgroup Command. 328
lsitab Command 330

lskbd Command . . . 330
lslicense Command . . . 331
lslpp Command . . . 332
lslv Command . . . 336
lsmaster Command . . . 339
lsmcode Command . . . 340
lsmksysb Command . . . 342
lsnamsv Command . . . 344
lsnfsexp Command . . . 345
lsnfsmnt Command . . . 346
lsnim Command . . . 347
lsnlspath Command . . . 350
lsparent Command . . . 350
lspath Command . . . 352
lsprtsv Command . . . 357
lsps Command . . . 358
lspv Command . . . 359
lsque Command . . . 362
lsquedev Command . . . 364
lsresource Command . . . 365
lsresponse Command . . . 368
lsrole Command . . . 372
lsrpdomain Command . . . 373
lsrpnode Command . . . 376
lsrset Command . . . 378
lsrsrc Command . . . 379
lsrsrcdef Command . . . 384
lssavevg Command . . . 389
lssec Command . . . 391
lssensor Command . . . 393
lsslot Command . . . 395
lssrc Command . . . 397
lstun Command . . . 399
lsuser Command . . . 400
lsvfs Command . . . 402
lsvg Command . . . 403
lsvirprt Command . . . 407
lsvmode Command . . . 410
lswlmconf Command . . . 411
lvmstat Command . . . 414
m4 Command . . . 416
mach Command . . . 419
machstat Command . . . 420
macref Command . . . 420
mail, Mail, or mailx Command . . . 422
mailq Command . . . 434
mailstats Command . . . 436
make Command . . . 437
makedbm Command . . . 443
makedepend Command . . . 444
makedev Command . . . 447
makekey Command . . . 447
makemap Command . . . 448
man Command . . . 449
managefonts Command . . . 452
mant Command . . . 454

mark Command 455
mesg Command 458
mgmtsvr Command 459
mhl Command 461
mhmail Command 463
mhpath Command. 464
migratelp Command 466
migratepv Command 467
mirrord Daemon 468
mirrorvg Command 470
mk_niscachemgr Command 473
mk_nisd Command 474
mk_nispasswdd Command 475
mkboot Command. 476
mkC2admin Command 478
mkcatdefs Command. 479
mkCCadmin Command 481
mkcd Command 482
mkcfsmnt Command 486
mkclass Command 487
mkclient Command 489
mkcomg Command 490
mkcondition Command 493
mkcondresp Command 498
mkdev Command 500
mkdir Command 503
mkdirhier Command 504
mkfifo Command 505
mkfilt Command 506
mkfont Command 507
mkfontdir Command 507
mkfs Command. 509
mkgroup Command 511
mkhosts Command 514
mkitab Command 515
mkinstallp Command. 517
mkkeyserv Command 518
mkkrb5clnt Command 519
mkkrb5srv Command 521
mklost+found Command 523
mklv Command. 523
mklvcopy Command 529
mkmaster Command. 531
mknamsv Command 532
mknetid Command 534
mknfs Command 535
mknfsexp Command 536
mknfsmnt Command 537
mknod Command 540
mknotify Command 541
mkpasswd Command 542
mkpath Command. 544
mkproto Command 546
mkprtldap Command. 550
mkprtsv Command 553
mkps Command 556

mkqos Command . . . 557
mkque Command . . . 558
mkquedev Command . . . 559
mkramdisk Command . . . 561
mkresponse Command . . . 563
mkrole Command . . . 567
mkrpdomain Command . . . 569
mkrset Command . . . 571
mkrsrc Command . . . 572
mkseckrb5 Command . . . 575
mksecldap Command . . . 576
mksecpki Command . . . 581
mksensor Command . . . 583
mkserver Command . . . 585
mkslave Command . . . 586
mkssys Command. . . . 587
mkstr Command . . . 590
mksysb Command . . . 591
mkszfile Command . . . 594
mktcpip Command . . . 595
mktun Command . . . 597
mkuser Command. . . . 597
mkuser.sys Command . . . 600
mkvg Command . . . 601
mkvgdata Command . . . 604
mkvirprt Command . . . 605
mm Command . . . 607
mmt Command . . . 609
mmtu Command . . . 611
mobip6ctrl Command. . . . 611
mobip6reqd Daemon . . . 613
monacct Command . . . 614
mon-cxma Command . . . 615
monitord Daemon . . . 616
moo Command . . . 617
more Command . . . 618
mosy Command . . . 622
mount Command . . . 624
mountd Daemon . . . 633
mpcfg Command . . . 634
mpcstat Command . . . 636
mrouted Daemon . . . 638
msgchk Command . . . 642
msh Command . . . 643
mt Command (BSD) . . . 644
mtrace Command . . . 646
mv Command . . . 648
mvdir Command . . . 650
mvfilt Command . . . 651
mvt Command . . . 651
mwm Command . . . 653

Appendix. Notices . . . 691
Trademarks . . . 692

Index . . . 693

About This Book

This book is Volume 3 of the six-volume *AIX 5L Version 5.2 Commands Reference*, which contains reference information on operating system commands. It describes the tasks each command performs, how commands can be modified, how they handle input and output, who can run them, and provides a master index for all six volumes.

For a quick reference list of commands arranged in functional groups, see Volume 6.

This edition supports the release of AIX 5L Version 5.2 with the 5200-01 Recommended Maintenance package. Any specific references to this maintenance package are indicated as *AIX 5.2 with 5200-01*.

Who Should Use This Book

This book is intended for users of operating system commands.

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.

© Copyright IBM Corp. 1997, 2003

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.

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.

The 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 UNIX98 Specification

Beginning with Version 4.3, the operating system is designed to support The Open Group's UNIX98 Specification 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 4.3 even more open and portable for applications.

At the same time, compatibility with previous releases of the operating system is preserved. This is accomplished by the creation of a new environment variable, which can be used to set the system environment on a per-system, per-user, or per-process basis.

To determine the proper way to develop a UNIX98-portable application, you may need to refer to The Open Group's UNIX98 Specification, which can be obtained on a CD-ROM by ordering *Go Solo 2: The Authorized Guide to Version 2 of the Single UNIX Specification*, a book which includes The Open Group's UNIX98 Specification on a CD-ROM.

Related Information

The following books contain information about or related to commands:

- *AIX 5L Version 5.2 Commands Reference, Volume 1*
- *AIX 5L Version 5.2 Commands Reference, Volume 2*
- *AIX 5L Version 5.2 Commands Reference, Volume 3*
- *AIX 5L Version 5.2 Commands Reference, Volume 4*
- *AIX 5L Version 5.2 Commands Reference, Volume 5*
- *AIX 5L Version 5.2 Commands Reference, Volume 6*
- *AIX 5L Version 5.2 Files Reference*
- *AIX 5L Version 5.2 Glossary*
- *AIX 5L Version 5.2 Guide to Printers and Printing*
- *AIX 5L Version 5.2 Installation Guide and Reference*
- *AIX 5L Version 5.2 AIX Installation in a Partitioned Environment*
- *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*
- *AIX 5L Version 5.2 Operating System Installation: Getting Started*
- *AIX 5L Version 5.2 Performance Management Guide*
- *AIX 5L Version 5.2 Performance Tools Guide and Reference*
- *AIX 5L Version 5.2 Security Guide*
- *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*
- *AIX 5L Version 5.2 System Management Guide: Communications and Networks*
- *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*

- *AIX 5L Version 5.2 System User's Guide: Communications and Networks*
- *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*
- *AIX 5L Version 5.2 Technical Reference: Base Operating System and Extensions Volume 1*
- *AIX 5L Version 5.2 Technical Reference: Base Operating System and Extensions Volume 2*
- *AIX 5L Version 5.2 Technical Reference: Communications Volume 1*
- *AIX 5L Version 5.2 Technical Reference: Communications Volume 2*
- *AIX 5L Version 5.2 Technical Reference: Kernel and Subsystems Volume 1*
- *AIX 5L Version 5.2 Technical Reference: Kernel and Subsystems Volume 2*
- *AIX 5L Version 5.2 Web-based System Manager Administration Guide*
- *Performance Toolbox Version 2 and 3 for AIX: Guide and Reference*

Alphabetical Listing of Commands

ibm3812 Command

Purpose

Postprocesses the **troff** command output for the IBM 3812 Model 2 Pageprinter.

Syntax

ibm3812 [**-altpaper**] [**-landscape**] [**-quietly**] [**-F***Directory*] [**-i**] [*File*...]

Description

The **ibm3812** command is a postprocessor that can be used on intermediate output produced by the **troff** command.

Note: An entire page is placed in memory before it is printed.

If given one or more file names as options, the **ibm3812** command processes those files. If no file names are specified, this command acts as a filter interpreting standard input.

The **ibm3812** command's font files allow the postprocessor to send characters of more than one byte to the printer. These can be characters that require multiple bytes to represent them, such as code page and point; or, they can be characters that are composed of two or more concatenated glyphs.

For example, the character code for the `\(ib` (improper subset) special character is:

```
"\001\125\xe2\xff\xe8\xe3%\x00\x16\001\074\xe3\xff\xea"
```

The printer is in Page Map Primitive (PMP) mode when these bytes are sent, so you must use the `001` directive to introduce a character. For single-byte codes, this Generic Font Patterns command is automatically handled by the postprocessor. The % (percent sign) characters escape the bytes containing 0, which would otherwise terminate the code sequence. To obtain a literal % character, escape it with another % character so that a percent sign is displayed as `%%`. A single-byte % code is assumed to be a literal percent sign, so that the single-byte % character needs no special handling in the font file.

Notes:

1. The **ibm3812** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm3812** directory. It does not produce usable output unless these files have been properly set up.
2. The postprocessor requires additional font information to be stored in the **/usr/lib/font/devibm3812/fonts** file. If new fonts are added to this file, make sure that the **DESC** file is also updated to reflect the additional fonts and special characters.

The format of the file must be preserved. The file contains the following four fields:

- The one- or two-letter name of the font
- The full name of the font on the printer-font diskette
- The one- or two-letter name of the substitute font
- An array of five available sizes.

Flags

-altpaper Specifies that the file should be printed from the alternate paper drawer. By default, the **ibm3812** command prints from the primary paper drawer.

© Copyright IBM Corp. 1997, 2003

-landscape	Specifies that the file should be printed in landscape orientation, so that the wider part of the paper is horizontally oriented. This flag rotates the page to the right by 90 degrees. By default, the **ibm3812** command prints in portrait orientation.
-quietly	Suppresses all non-fatal error messages.
-F*Directory*	Specifies the directory holding the font files. The default file is **devibm3812**. The command looks for font files in the **/usr/lib/font** directory by default.
-i	Suppresses initialization of the printer that runs the PMP.init macro, after the job has printed.

Example

Following is an example of the **troff** command used with the **ibm3812** command:

```
troff file|ibm3812|qprt-dp
```

Files

/usr/lib/font/devibm3812/*.out	Contains font files for the **ibm3812** command.
/usr/lib/font/devibm3812/fonts	Contains information about the available fonts for the **ibm3812** command.

Related Information

The **ibm3816** command, **troff** command.

The **troff** font file format specifies description files for the troff command.

---

ibm3816 Command

Purpose

Postprocesses the **troff** command output for the IBM 3816 Pageprinter.

Syntax

ibm3816 [**-altpaper**] [**-landscape**] [**-quietly**] [**-F***Directory*] [**-i**] [*File*...]

Description

The **ibm3816** command is a postprocessor that can be used on intermediate output produced by the **troff** command.

> **Note:** An entire page is placed in memory before it is printed.

If given one or more file names as options, the **ibm3816** command processes those files. If no file names are specified, this command acts as a filter interpreting standard input.

The **ibm3816** command's font files allow the postprocessor to send characters of more than one byte to the printer. These can be characters that require multiple bytes to represent them, such as code page and point; or, they can be characters that are composed of two or more concatenated glyphs.

For example, the character code for the `\(ib` (improper subset) special character is:

```
"\001\125\xe2\xff\xe8\xe3%\x00\x16\001\074\xe3\xff\xea"
```

The printer is in Page Map Primitive (PMP) mode when these bytes are sent, so you must use the `001` directive to introduce a character. For single-byte codes, this Generic Font Patterns command is automatically handled by the postprocessor. The % (percent sign) characters escape the bytes containing 0, which would otherwise terminate the code sequence. To obtain a literal % character, escape it with

another % character so that a percent sign is displayed as %%. A single-byte % code is assumed to be a literal percent sign, so that the single-byte % character needs no special handling in the font file.

Notes:

1. The **ibm3816** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm3816** directory. It does not produce usable output unless these files have been properly set up.
2. The postprocessor requires additional font information to be stored in the **/usr/lib/font/devibm3816/fonts** file. If new fonts are added to this file, make sure that the **DESC** file is also updated to reflect the additional fonts and special characters.

The format of the file must be preserved. The file contains the following four fields:

- The one- or two-letter name of the font
- The full name of the font on the printer-font diskette
- The one- or two-letter name of the substitute font
- An array of five available sizes.

Flags

-altpaper	Specifies that the file should be printed from the alternate paper drawer. By default, the **ibm3816** command prints from the primary paper drawer.
-landscape	Specifies that the file should be printed in landscape orientation, so that the wider part of the paper is horizontally oriented. This flag rotates the page to the right by 90 degrees. By default, the **ibm3816** command prints in portrait orientation.
-quietly	Suppresses all non-fatal error messages.
-F*Directory*	Specifies the directory holding the font files. The default file is **devibm3816**. The command looks for font files in the **/usr/lib/font** directory by default.
-i	Suppresses initialization of the printer that runs the PMP.init macro, after the job has printed.

Example

Following is an example of the **troff** command used with the **ibm3816** command:

```
troff file|ibm3816|qprt-dp
```

Files

/usr/lib/font/devibm3816/*.out	Contains font files for the **ibm3816** command.
/usr/lib/font/devibm3816/fonts	Contains information about the available fonts for the **ibm3816** command.

Related Information

The **ibm3812** command, **troff** command.

The **troff** font file format specifies description files for the troff command.

---

ibm5585H-T Command

Purpose

Processes **troff** command output for the IBM 5585H-T printer.

Syntax

ibm5585H-T [**-F***Directory*] [*File*]

Description

The **ibm5585H-T** command processes the output of the **troff** command for output to the IBM 5585H-T printer for traditional Chinese language. This command is provided exclusively for traditional Chinese language support.

The **ibm5585H-T** command processes one or more files specified by the *File* parameter. If no file is specified, the **ibm5585H-T** command reads from standard input.

The **ibm5585H-T** command uses font files in the **/usr/lib/font/devibm5585H-T** directory that have command names ending with **.out**. The **ibm5585H-T** command does not produce correct output unless these files are provided.

Flag

-F*Directory* Specifies a directory name as the place to find font files. By default, the **ibm5585H-T** command looks for font files in the **/usr/lib/font/devibm5585H-T** directory.

Example

To process the `reports` file for the IBM 5585H-T printer, enter:

```
troff reports |ibm5585H-T | qprt -dp
```

The **ibm5585H-T** command first processes the output of the **troff** command, then sends the file to a print queue.

File

/usr/lib/font/devibm5585H-T/*.out Contains font files.

Related Information

The **troff** command.

The **troff** font file format.

---

ibm5587G Command

Purpose

Postprocesses **troff** command output for the IBM 5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 printers with the (32x32/24x24) cartridge installed. This command is used exclusively for Japanese Language Support.

Syntax

ibm5587G [**-F***Directory*] [**-quietly**] [*File* ...]

Description

The **ibm5587G** command processes the output of the **troff** command for output to the 5587-G01, 5584-H02, 5585-H01, 5587-H01, and 5589-H01 printers.

If given one or more files as options, the **ibm5587G** command processes those files. If no files are specified, it acts as a filter interpreting standard input.

> **Note:** The **ibm5587G** command assumes that the (32x32/24x24) cartridge is installed in the printer. Incorrect output from the printer may result if the wrong cartridge is installed in the printer.

The **ibm5587G** command depends on the files with names ending in **.out** in the **/usr/lib/font/devibm5587G** directory. It does not produce reasonable output unless these files have been properly set up.

Flags

-F*Directory*	Specifies a directory name as the place to find the font files. By default, the **ibm5587G** command looks for font files in the **/usr/lib/font/devibm5587G** directory.
-quietly	Suppresses all nonfatal error messages.

Files

/usr/lib/font/devibm5587G/*.out	Contains font files.

Related Information

The **troff** command formats text for printing on typesetting devices.

The **troff** font file format specifies description files for the troff command.

---

iconv Command

Purpose

Converts the encoding of characters from one code page encoding scheme to another.

Syntax

iconv -f *FromCode* **-t** *ToCode* [*FileName*...]

Description

The **iconv** command converts the encoding of characters read from either standard input or the specified file from one coded character set to another and then writes the results to standard output. The input and output coded character sets are identified by the *FromCode* and *ToCode* parameters. The input data should consist of characters in the code set specified by the *FromCode* parameter. If the *FileName* parameter is not specified on the command line, the **iconv** command reads from standard input.

You can use a Web-based System Manager System application (**wsm system** fast path) to run this command. You could also use the System Management Interface Tool (SMIT) **smit iconv** fast path to run this command.

Flags

-f *FromCode*	Specifies the code set in which the input data is encoded. The space between the **-f** flag and the *FromCode* parameter is optional.
-t *ToCode*	Specifies the code set to which the output data is to be converted. The space between the **-t** flag and the *ToCode* parameter is optional.
FileName	Specifies a file to be converted.

The list of supported code set converters is provided in ″List of Converters″ in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Exit Status

This command returns the following exit values:

0 Input data was successfully converted.
1 The specified conversions are not supported; the given input file cannot be opened for read; or there is a usage-syntax error.
2 An unusable character was encountered in the input stream.

Examples

1. To convert the contents of the **mail.x400** file from code set IBM-850 and store the results in the **mail.local** file, enter:

   ```
   iconv -f IBM-850 -t ISO8859-1 mail.x400 > mail.local
   ```

2. To convert the contents of the **mail.japan** file from the 7-bit interchange (ISO2022) encoding to the Japanese EUC code set (IBM-eucJP), enter:

   ```
   iconv -f fold7 -t IBM-eucJP mail.japan > mail.local
   ```

3. To convert the contents of a local file to the mail-interchange format and send mail, enter:

   ```
   iconv -f IBM-943 -t fold7 mail.local | mail fxrojas
   ```

Related Information

The **genxlt** command describes how to define a conversion table.

The **iconv** subroutine, **iconv_close** subroutine, and **iconv_open** subroutines provide a method to use the conversion service from within a program.

Converters Overview in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Converters Overview for Programming in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

---

id Command

Purpose

Displays the system identifications of a specified user.

Syntax

id [{ **-G** | **-g** [**-r**] | **-u** [**-r**] } [**-n**]] [*User*]

Description

The **id** command writes to standard output a message containing the system identifications (ID) for a specified user. The system IDs are numbers which identify users and user groups to the system. The **id** command writes the following information, when applicable:

- User name and real user ID
- Name of the user's group and real group ID
- Name of user's supplementary groups and supplementary group IDs

Supplementary group information is written only for systems supporting multiple-user groups and only if the specified user belongs to a supplementary group.

The **id** command also writes effective user and group IDs, but only for the user that invoked the **id** command. (If the *User* parameter is specified with the **id** command, the effective IDs are assumed to be identical to real IDs.) If the effective and real IDs for the invoking user are different, the **id** command writes the following effective ID information, when applicable:

- Effective user name and effective user ID
- Name of effective user's group and effective group ID

The **id** command will fail if the specified user does not exist or if the command cannot read the user or group information.

Flags

The contents and format of the message written by the **id** command can be altered with the following flags:

-G Specifies that the **id** command write the effective, real, and supplementary group IDs only. If there are multiple entries for the effective, real, or supplementary IDs, they are separated by a space and placed on the same line.

-g Specifies that the **id** command write only the effective group ID.

-u Specifies that the **id** command write only the effective user ID.

-r Specifies that the **id** command write the real ID instead of the effective ID. This flag can be invoked with either the **-g** flag to write the real group ID, or the **-u** flag to write the real user ID.

-n Specifies that the **id** command outputs the name, instead of the ID number, when it is specified with the **-G**, **-g**, and **-u** flags.

User Specifies the login name of a user for the **id** command. If no user is specified, the user invoking the **id** command is the default.

Security

Access Control: This program should be installed as a normal user program in the Trusted Computing Base.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To display all system identifications for the current user, enter:

   ```
   id
   ```

   Output for the **id** command is displayed in the following format:

   ```
   uid=1544(sah) gid=300(build) euid=0(root) egid=9(printq) groups=0(system),10(audit)
   ```

   In this example, the user has user name `sah` with an ID number of `1544`; a primary group name of `build` with an ID number of `300`; an effective user name of `root` with an ID number of `0`; an effective group name of `printq` with an ID number of `9`; and two supplementary group names of `system` and `audit`, with ID numbers `0` and `10`, respectively.

2. To display all group ID numbers for the current user, enter:

   ```
   id -G
   ```

Output is displayed in the following format:

```
0 10 300 9
```

The **-G** flag writes only the group IDs for a user. In this example, user `sah` is a member of the system (`0`), audit (`10`), build (`300`), and printq (`9`) groups.

3. To display all group names for the current user, enter:

```
id -Gn
```

Output is displayed in the following format:

```
system audit build printq
```

The **-n** flag writes only the names instead of the ID numbers.

4. To display the real group name for the current user, enter:

```
id -gnr
```

Output is displayed in the following format:

```
build
```

Files

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

Related Information

The **getty** command, **login** command, **setgroups** command, **su** command, **tsm** command.

---

ifconfig Command

Purpose

Configures or displays network interface parameters for a network using TCP/IP.

Syntax

ifconfig *Interface* [*AddressFamily* [*Address* [*DestinationAddress*]] [*Parameters*...]]

ifconfig *Interface* [*ProtocolFamily*] *Interface ProtocolFamily*

ifconfig -a [**-l**] [**-d**] [**-u**] [*ProtocolFamily*]

Description

You can use the **ifconfig** command to assign an address to a network interface and to configure or display the current network interface configuration information. The **ifconfig** command must be used at system startup to define the network address of each interface present on a machine. After system startup, it can also be used to redefine an interfaces address and its other operating parameters. The network interface configuration is held on the running system and must be reset at each system restart. The **ifconfig** command interprets the **IFF_MULTICAST** flag and prints its value if it is set.

An interface can receive transmissions in differing protocols, each of which may require separate naming schemes. It is necessary to specify the *AddressFamily* parameter, which may change the interpretation of the remaining parameters. The address families currently supported are **inet**, **inet6**, and **ns**.

For the DARPA-Internet family, **inet**, the address is either a host name present in the host name database, that is, the **/etc/hosts** file, or a DARPA-Internet address expressed in the Internet standard dotted decimal notation.

For the Xerox Network Systems (XNS) family, **ns**, addresses are *net:a.b.c.d.e.f.*, where *net* is the assigned network number (in decimals), and each of the 6 bytes of the host number, `a` through `f`, are specified in hexadecimal. The host number may be omitted on 10-Mbps Ethernet interfaces, which use the hardware physical address, and on interfaces other than the first interface.

While any user can query the status of a network interface, only a user who has administrative authority can modify the configuration of those interfaces.

The **ifconfig** function displays the current configuration for a network interface when no optional parameters are supplied.

If a protocol family is specified, **ifconfig** will report only the details specific to that protocol family.

Only a super user may modify the configuration of a network interface.

Gratuitous ARP is supported for Ethernet, token-ring, and FDDI interfaces. This means when an IP address is assigned, the host sends an ARP request for its own address (the new address) to inform other machines of its address so that they can update their ARP entry immediately. It also lets hosts detect duplicate IP address. If you get a response to the ARP request, an error is logged in **/var/adm/ras/errlog** which can be viewed using **errpt** command (or using SMIT interface) for the error ID **AIXIF_ARP_DUP_ADDR**.

Flags

-a	Optionally, the **-a** flag may be used instead of an interface name. This flag instructs **ifconfig** to display information about all interfaces in the system.
-d	The **-d** flag displays interfaces that are down.
-l	This flag may be used to list all available interfaces on the system, with no other additional information. Use of this flag is mutually exclusive with all other flags and commands, except for **-d** and **-u**.
-u	The **-u** flag displays interfaces that are up.
ProtocolFamily	This flag specifies protocols such as **tcp**, **udp**, **tcp6**, **udp6**, **icmp**, and **icmp6**.

Parameters

Address	Specifies the network address for the network interface. For the **inet** family, the *Address* parameter is either a host name or an Internet address in the standard dotted decimal notation.
AddressFamily	Specifies which network address family to change. The **inet**, **inet6**, and **ns** address families are currently supported. This parameter defaults to the **inet** address family.
DestinationAddress	Specifies the address of the correspondent on the remote end of a point-to-point link.

Interface

Specifies the network interface configuration values to show or change. You must specify an interface with the *Interface* parameter when you use the **ifconfig** command. Abbreviations for the interfaces include:

- at for ATM (Asynchronous Transfer Mode)
- en for Standard Ethernet (inet, xns)
- et for IEEE 802.3 Ethernet (inet, xns)
- tr for Token-Ring (inet, xns)
- xt for X.25 (inet)
- sl for serial line IP (inet)
- lo for loopback (inet)
- op for serial (inet)
- vi for Virtual IP Address (inet)

Include a numeral after the abbreviation to identify the specific interface (for example, `tr0`).

If *Interface* is not yet loaded, **ifconfig** *Interface* loads that interface and **netstat -in** lists it. In processing a status query for *Interface*, that interface is loaded (if not already loaded) to complete the query processing.

Parameter

Allows the following parameter values:

alias Establishes an additional network address for the interface. When changing network numbers, this parameter is useful for accepting packets addressed to the old interface.

allcast Sets the Token-Ring interface to broadcast to all rings on the network.

-allcast
Confines the Token-Ring interface to broadcast only to the local ring.

arp Enables the **ifconfig** command to use the Address Resolution Protocol in mapping between network-level addresses and link-level addresses. The **arp** value is the default.

-arp Disables the use of the Address Resolution Protocol.

authority
Reserved for future use.

bridge Reserved for future use.

-bridge Reserved for future use.

broadcast *Address*
(**inet** only) Specifies the address to use to broadcast to the network. The default broadcast address has a host part of all 1s.

-dad (inet6 only) Does not perform duplicate IPv6 address detection.

-debug Disables driver-dependent debug code.

delete Removes the specified network address. This is used when an alias is incorrectly specified or when it is no longer needed. Incorrectly setting an **ns** address has the side effect of specifying the host portion of the network address. Removing all **ns** addresses allows you to re-specify the host portion.

device *dev_name*
This parameter applies to ATM Network interface only. It specifies the device name this interface is associated with. Unlike Token Ring or Ethernet, in case of ATM, there is not a one-to-one correspondence between interface and device. In the case of ATM, there can be more than one interface for every device.

detach Removes an interface from the network interface list. If the last interface is detached, the network interface driver code is unloaded. In order for the interface route of an attached interface to be changed, that interface must be detached and added again with **ifconfig**.

down Marks an interface as inactive (**down**), which keeps the system from trying to transmit messages through that interface. If possible, the **ifconfig** command also resets the interface to disable reception of messages. Routes that use the interface, however, are not automatically disabled.

eui64 (inet6 only) The real IPv6 address is computed by replacing the last 64 bytes of the given address with the Interface Identifier.

first Puts an IPv6 address at the first place on an interface in order to select it as the source for unbound sockets. The syntax for using this parameter is,

```
ifconfig interface inet6 first address
```

firstalias
(inet6 only) Same as alias, but sets the address in front of the interface address list in order to select it as the source for unbound sockets.

group *ID*
Adds a group ID to the group ID list for the interface. This list is used in determining the route to use when forwarding packets that arrived on the interface.

-group *ID*
Removes a group ID from the group ID list for the interface. This list is used in determining the route to use when forwarding packets that arrived on the interface.

hwloop
Enables hardware loopback. The hardware loopback specifies that locally addressed packets handled by an interface should be sent out using the associated adapter.

-hwloop
Disables hardware loop-back. The hardware loop-back specifies that locally addressed packets handled by an interface should be sent out using the associated adapter.

ipdst Specifies an Internet host willing to receive IP packets encapsulating **ns** packets bound for a remote network. An apparent point-to-point link is constructed, and the specified address is taken as the **ns** address and network of the destination.

ipv6dst
Used to specify an IPv6 node that is willing to receive IPv6 packets encapsulating IPv6 or IPv4 packets through a tunnel. The apparent destination of the point to point tunnel interface may not be the real destination of the packets. At the tunnel endpoint, the decapsulated packets may then be forwarded to their final destination.

link [0-2]
Enables special processing of the link level of the interface. These three options are interface-specific. In actual effect, however, they are generally used to select special modes of operation. An example of this is to enable SLIP compression, or to select the connector type for some Ethernet cards. Refer to the manual page for the specific driver for more information.

-link [0-2]
Disables special processing at the link level with the specified interface.

metric *Number*
Sets the routing metric of the interface to the value specified by the *Number* variable. The default is 0 (zero). The routing metric is used by the routing protocol (the **routed** daemon). Higher metrics have the effect of making a route less favorable. Metrics are counted as addition hops to the destination network or host.

mtu *Value*
Sets the maximum IP packet size for this system. The *Value* variable can be any number from 60 through 65535, but is media dependent. See ″Automatic Configuration of Network Interfaces″ in *AIX 5L Version 5.2 System Management Guide: Communications and Networks* for maximum transmission unit (MTU) values by interface.

netmask *Mask*
Specifies how much of the address to reserve for subdividing networks into subnetworks. This parameter can be used only with an address family of **inet**.

The *Mask* variable includes both the network part of the local address and the subnet part, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number beginning with 0x, in standard Internet dotted decimal notation, or beginning with a name or alias that is listed in the **/etc/networks** file.

In the 32-bit address, the mask contains 1s (ones) for the bit positions reserved for the network and subnet parts and 0s for the bit positions that specify the host. The mask should contain at least the standard network portion, and the subnet segment should be contiguous with the network segment.

pvc This parameters applies to ATM Network interface only. It specifies that this interface will support PVC (Permanent Virtual Circuit) types of virtual connections only.

pktchain
Enables the flag to indicate that this interface can handle multiple packets chained together on the output path.

-pktchain
Disables the flag that indicates that this interface can handle multiple packets chained together on the output path.

svc_c *server_addr*
This parameter applies to ATM Network interface only. It specifies that this interface will support both SVC (Switched Virtual Circuit) and PVC types of virtual connections. It further specifies that this interface will be an ARP client. The *server_addr* is the list of 20 byte ATM address of the ARP servers that this client will use. The addresses are specified in the form of xx.xx....xx. The first entry is considered the Primary ARP server and the rest are considered Secondary ARP servers. The list of 20 byte ARP server addresses must separated by a comma.

site6 Sets the IPv6 site number (default is zero). This should be used only with site-local addresses on a multi-sited node.

svc_s This parameter applies to ATM Network interface only. It specifies that this interface will support both SVC and PVC types of virtual connections. It further specifies that this interface will be the ARP server for this Logical IP Subnetwork (LIS).

security
Reserved for future use.

snap Reserved for future use.

-snap Reserved for future use.

tcp_nocksum
Disables verification of the checksum of TCP data for local traffic to the subnet attached to the interface. Checksum verification of TCP, UDP and IP headers continues. Checksum verification of TCP data read or written from this interface from or to remote networks also continues.

-tcp_nocksum
Enables verification of the checksum of TCP data for local traffic to the subnet attached to the interface. This is the default.

thread (**inet** only) Configures dedicated kernel threads for an interface. This parameter can only be used on SMP machines that have multiple CPU's. This parameter causes input packets to be queued to a kernel thread after processing by the device driver and input demuxer. The input packet is processed in IP and TCP or UDP by the thread instead of directly on the interrupt level. Setting this parameter can improve throughput when high speed adapters bottleneck on a single CPU during interrupt processing by allowing the input packets to be processed on other CPU's running the kernel threads (improved pipelining). For some work loads, this parameter increases the per packet overhead, due to thread scheduling overhead, resulting in higher CPU utilization an possibly lower throughput. This parameter only applies to AIX 4.3.3 or later.

-thread (**inet** only) Configures Disables kernel thread support that has been configured with the *thread* parameter. This parameter only applies to AIX 4.3.3 or later.

up Marks an interface as active (**up**). This parameter is used automatically when setting the first address for an interface. It can also be used to enable an interface after an **ifconfig down** command.

vipa_iflist
Adds the interfaces to the list of interfaces that should use this **vipa** as the source address in the outgoing packets.

-vipa_iflist
Removes the interfaces from the list of interfaces that are configured to use this **vipa** as the source address in the outgoing packets.

In AIX 4.3.3 and later versions, the following network options, commonly known as ISNO (Interface Specific Network Options), can be configured on a per interface basis:

rfc1323 [0 | 1]
Enables or disables TCP enhancements as specified by RFC 1323, *TCP Extensions for High Performance*. A value of 1 specifies that all TCP connections using this interface will attempt to

negotiate the RFC enhancements. A value of 0 disables **rfc1323** for all connections using this interface. The SOCKETS application can override this ISNO and global behavior on individual TCP connections with the **setsockopt** subroutine.

-rfc1323
Removes the use of ISNO for **rfc1323** for this network. A SOCKETS application can override the global behavior on individual TCP connections using the **setsockopt** subroutine.

tcp_mssdflt *Number*
Sets the default maximum segment size used in communicating with remote networks. If communicating over this interface, a socket uses *Number* as the value of the default maximum segment size.

-tcp_mssdflt
Removes the use of ISNO for **tcp_mssdflt**. The global value, manipulated via **/usr/sbin/no**, is used instead.

tcp_recvspace *Size*
Specifies the default socket buffer *Size* for interface sockets receiving data. The buffer size affects the window size used by TCP. (See the **no** command for more information.)

-tcp_recvspace
Removes the use of ISNO for **tcp_recvspace**. The global value is used instead.

tcp_sendspace *Size*
Specifies the default socket buffer *Size* for interface sockets sending data. The buffer size affects the window size used by TCP. (See the **no** command for more information.)

-tcp_sendspace
Removes the use of ISNO for **tcp_sendspace**. The global value is used instead.

tcp_nodelay [0 | 1]
Specifies that sockets using TCP over this interface follow the Nagle algorithm when sending data. By default, TCP follows the Nagle algorithm.

-tcp_nodelay
Removes the use of ISNO for the **tcp_nodelay** option.

Note: ISNO parameters set by **ifconfig** are lost at the next reboot. Use the **chdev** command to change the ODM database for each interface if you wish to make the ISNOs permanent. Use `lsdev -E -l [interface]` to see the interface attributes and `chdev -l -a [attribute=value]` to change the desired attribute. For example:

```
lsattr -E -l en0
chdev -l en0 -a tcp_sendspace=65536
```

Examples

The following are examples using the **ifconfig** command on a TCP/IP network and an XNS network:

Inet Examples

1. To query the status of a serial line IP interface, enter the command in the following format:

   ```
   ifconfig  sl1
   ```

   In this example, the interface to be queried is `sl1`. The result of the command looks similar to the following:

   ```
   sl1:  flags=51<UP,POINTOPOINT,RUNNING>
              inet  192.9.201.3  -->  192.9.354.7  netmask  ffffff00
   ```

2. To configure the local loop-back interface, enter the command in the following format:

   ```
   ifconfig  lo0  inet  127.0.0.1  up
   ```

3. To mark the local Token-Ring interface as down, enter the command in the following format:

```
ifconfig  tr0  inet  down
```

In this example, the interface to be marked is `token0`.

Note: Only a user with root user authority can modify the configuration of a network interface.

4. To turn **rfc1323** off for all connections over `en5` (assuming that the global value is 1), enter:
   ```
   ifconfig en0 rfc1323 0
   ```
5. To configure a list of interfaces to use a vipa, enter:
   ```
   ifconfig vi0 vipa_iflist en0,en1,tr0
   ```
6. To remove interface(s) that are configured to use vipa, enter:
   ```
   ifconfig vi0 -vipa_iflist en1,tr0
   ```
7. To find out which interfaces are configured to use a vipa, say vi0, enter:
   ```
   ifconfig vi0
   ```

XNS Examples

1. To configure a Standard Ethernet type interface for XNS, enter the command in the following format:
   ```
   ifconfig  en0  ns  110:02.60.8c.2c.a4.98  up
   ```

   In this example, `ns` is the XNS address family, `110` is the network number and `02.60.8c.2c.a4.98` is the host number, which is the Ethernet address unique to each individual interface. Specify the host number when there are multiple Ethernet hardware interfaces, as the default may not correspond to the proper interface. The Ethernet address can be obtained by the commands:
   ```
   ifconfig  en0
   netstat  -v
   ```

   The XNS address can be represented several ways, as shown in the following examples:
   - `123#9.89.3c.90.45.56`
   - `5-124#123-456-900-455-749`
   - `0x45:0x9893c9045569:90`
   - `0456:9893c9045569H`

   The XNS address in the first example is in decimal format. The address in the second example is broken into groups of three digits, with each group separated by a - (minus sign). The `0x` and `H` addresses are in hex format. Finally, the `0` in front of the last address indicates that the number is in octal format.
2. To configure an IEEE Ethernet 802.3 type interface for XNS, enter the command in the following format:
   ```
   ifconfig  et0  ns  120:02.60.8c.2c.a4.98  up
   ```

   The `en0` and `et0` interfaces are considered as separate interfaces even though the same Ethernet adapter is used. Two separate networks can be defined and used at the same time as long as they have separate network numbers. Multiple Ethernet adapters are supported.

   Note: The host number should correspond to the Ethernet address on the hardware adapter. A system can have multiple host numbers.
3. To configure an Internet encapsulation XNS interface, enter the command in the following format:
   ```
   ifconfig  en0  inet  11.0.0.1  up
   ifconfig  en0  ns  110:02.60.8c.2c.a4.98  up
   ifconfig  en0  ns  130:02.60.8c.34.56.78  ipdst  11.0.0.10
   ```

   The first command brings up the Internet with the `inet` address `11.0.0.1`. The second command configures the `en0` interface to be network `110` and host number `02.60.8c.2c.a4.98` in the `ns` address family. This defines the host number for use when the XNS packet is encapsulated within the Internet

packet. The last command defines network 130, host number 02.60.8c.34.56.78, and destination Internet address 11.0.0.10. This last entry creates a new network interface, nsip. Use the **netstat -i** command for information about this interface.

Files

/etc/host	Contains the host-name database.
/etc/networks	Contains network names.

Related Information

The **netstat** command.

The **hosts** file format, **networks** file format.

TCP/IP Network Interfaces, Understanding Protocols, TCP/IP Routing, Subnet Addresses in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Xerox Network Systems (XNS) Overview for Programming in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

---

ike Command

Purpose

Starts, stops, and monitors IP Security dynamic tunnels which use the Internet Key Exchange Protocol (ISAKMP/Oakley).

Syntax

ike cmd=*Subcommand* [*parameter* ...]

Description

The **ike** is used to start, stop, and monitor IP Security dynamic tunnels using the Internet Key Exchange (IKE) protocol. IP Security tunnels protect IP traffic by authenticating and/or encrypting IP data. The **ike** command performs several functions. It can activate, remove, or list IKE and IP Security tunnels. For an overview of IP Security and IKE tunnels, see IP Security in the *AIX 5L Version 5.2 Security Guide*.

Note: You must have root access to use the **ike** command.

The IKE negotiation occurs in two phases. The first phase authenticates the two parties and sets up a **Key Management** (also known as phase 1) **Security Association** for protecting the data that is passed during the negotiation. In this phase the key management policy is used to secure the negotiation messages. The second phase negotiates **Data Management** (also known as the phase 2) **Security Association**, which uses the data management policy to set up IP Security tunnels in the kernel for encapsulating and decapsulating data packets. The secure channel established in phase 1 can be used to protect multiple data management negotiations between 2 hosts.

The **ike** command is used to activate tunnels with identification and policy information which has already been entered using the **ikedb** command or the Web-based System Manager Graphical User Interface (GUI) under Virtual Private Networks (IP Security) in the Network application. The parameters to be used during the negotiation are entered by the user and stored in a database. The **ike** command allows the activation, removal and listing of tunnels that have been started using the security parameters stored in the database.

In most uses of the **ike** command, activation and deletion occurs for both phases, however the command allows these operations to be done separately.

Subcommands

activate

Purpose

Start the negotiation of an IKE tunnel. If phase is not specified, both a phase 1 and phase 2 tunnel are started. If IP addresses are supplied, the tunnel is setup using those IP addresses. If the IDs used during the negotiation are not IP addresses, the local and remote host IDs must be entered using the Virtual Private Networks Web-based System Manager Graphical User Interface (GUI) panels. A unique tunnel number is created. The tunnel can then be referenced by the tunnel number in the **ike** command to indicate the particular tunnel to be started.

Syntax

ike cmd=activate [**phase=1**|**2**] [**numlist=***tunnel_num_list*] [**namelist=tunnel_name_list**] [**remid=***remote_id*] [**ipaddr=***src_addr*,*dst_addr*] [**autostart**]

Description

The **activate** subcommand works using a two phase paradigm. A phase 1 tunnel must be established before a phase 2 tunnel can be started. If a phase 1 tunnel is specified, then only the phase 1 tunnel negotiation takes place. If a phase 2 tunnel is specified, the system checks for the existence of the corresponding phase 1 tunnel before creating the phase 2 tunnel. If the phase 1 negotiation has not been started, it is started automatically.

Upon successful completion of a phase 2 tunnel, the tunnel definition and corresponding filter rules are inserted into the IP Security kernel, and the new tunnel is activated. Traffic described by the tunnel definition passing between the designated endpoints is protected by the encryption and authentication algorithms indicated by the associated IKE security policy.

Multiple phase 2 tunnels can be started under the same phase 1 tunnel. A situation where this may be desired is if different types of traffic between two endpoints need different levels of security protection. The Security Association used for the phase 1 tunnel can be shared by multiple phase 2 tunnels. The phase 2 tunnels would specify the type of traffic (by protocol and port, or subnet mask, for instance) and could have different security policies protecting them.

The **ike** command returns if either a negotiation has been initiated, an error returns, or the tunnel already exists. Since the remote host must be contacted during the negotiation and the amount of time needed to complete the negotiation is uncertain, the **list** subcommand should be used to determine if the negotiation was successful.

Errors that are detected during the negotiation process can be captured by using **syslog**.

Flags

phase Specifies the type of negotiation desired. If omitted, the **activate** subcommand activates both a phase 1 and phase 2 tunnel. The **phase** flag is an optional flag.

numlist
Initiates the **ike** tunnel number which corresponds to the desired phase 1 or phase 2 tunnel(s) to be started. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. The **list** subcommand with the database option **db** can be used to determine the tunnel number for a particular tunnel. An example using tunnel numbers is shown below:

```
ike cmd=activate numlist=1,3,5-7
```

This would start tunnels 1, 3, 5, 6 and 7.

remid Starts phase 1 or phase 2 tunnel(s) from the local ID to the specified remote ID. **remid** could be a phase 1 ID (such as IP address, FQDN, user FQDN and X500DN), a phase 2 ID (such as IP address, subnet and IP address range) or a group ID. The **,** (comma) is used to delimit the subnet id and subnet mask, and the starting and ending IP address. If **remid** is a group name, a tunnel is initiated for each group member. **remid** is an optional flag and can only be used with the **activate** subcommand. It cannot be used in conjunction with the **ipaddr**, **numlist** or **namelist** flags.

1. To activate a phase 1 tunnel to remote IP address 9.3.97.100, type:

   ```
   ike cmd=activate phase=1 remid=9.3.97.100
   ```

2. To activate a phase 2 tunnel to remote subnet ID 9.3.97.100,255.255.255.0, type:

   ```
   ike cmd=activate phase=2 remid=9.3.97.100,255.255.255.0
   ```

ip_addr
Starts a phase 1 or phase 2 tunnel between the specified IP Addresses.

autostart
Causes the activation of all phase 1 and phase 2 tunnel database entries which were created with the **autostart** parameter set. The **autostart** flag does not work in conjunction with any other flags pertaining to the **activate** subcommand.

namelist
Specifies a tunnel name or comma-separated list of tunnel names to be activated. This flag requires the use of the **phase** flag.

Examples

1. To activate a phase 2 tunnel between source IP address x.x.x.x and destination IP address y.y.y.y, enter:

   ```
   ike cmd=activate phase=2 ipaddr=x.x.x.x,y.y.y.y
   ```

   The security policy indicated in the database for the IP addresses x.x.x.x and y.y.y.y is used for activating the tunnel.
2. To activate phase 1 tunnels for tunnels 1 and 2, enter:

   ```
   ike cmd=activate phase=1 numlist=1,2
   ```

3. To activate phase 2 tunnels for inactive tunnels named AIXFW1_DM and remote_office in the database enter:

   ```
   ike cmd=activate phase=2 namelist=AIXFW1_DM,remote_office
   ```

   Note: Because each phase 2 tunnel must have an associated phase 1 tunnel, a phase 1 tunnel is automatically activated before the phase 2 tunnel is activated.

list

Purpose

Monitors the status of IP Security tunnels by phase. It is also used to view tunnel entries defined in the IKE database.

Syntax

ike cmd=list [**phase=1**|**1+**|**2**] [**numlist=** *tunnel_num_list*] [**db** | **role=i**|**r**] [**verbose**]

Description

The **list** subcommand queries the Tunnel Manager and lists phase 1 and phase 2 tunnel status and information according to the result of the query. This command can also be used to view information in the Tunnel Definition database. The default behavior is to list the tunnels currently active. To list the tunnels in the database, the **db** option must be used.

Flags

phase Indicates the type and order of the tunnel(s) to be listed. A phase value of 1 results in only the requested phase 1 tunnel information being displayed. A phase value of 2 results in the information for the requested phase 2 tunnel(s) and their associated phase 1 tunnel(s) should be displayed. A phase value of 1+ means that the requested phase 1 tunnel and all associated phase 2 tunnels should be displayed. The default phase value is 1+.

numlist

Lists of the tunnel numbers which you would like to view. If omitted, the information from all tunnels is displayed. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. For example:

```
ike cmd=list numlist=1,3,5-7
```

When used in conjunction with **db**, tunnels from the IKE Security Policy database are shown.
Note: Active tunnel numbers and tunnel numbers from the IKE Tunnel Definitions database do not necessarily match up. This is because a single tunnel entry in the database can correspond to multiple active tunnels.

db Shows the entries in the database. If this flag is omitted, only active tunnels are displayed. This cannot be used in conjunction with **role**. Supply the list of tunnel numbers which you would like to view.

role Allows the display of tunnels by the point of initiation. If **i** is specified, then the tunnels that were initiated by the local host are displayed. If **r** is specified, then the tunnels where the local host acted as a responder are displayed. If this flag is omitted, both initiator and responder tunnels are shown. This flag cannot be used in conjunction with **db**.

verbose

Shows extended information about the specified tunnels. If this flag is not specified, then only a concise entry for each tunnel is shown.

Examples

Note: Tunnel numbers from the database and tunnel numbers from the tunnel manager do not necessarily reflect the same tunnel.

1. To perform a concise (short form) listing of phase 1 tunnels with entries in the tunnel manger, enter:

   ```
   ike cmd=list phase=1 numlist=1,2,3
   ```

   These tunnels are either being negotiated, in the active state , or have expired. Only tunnels 1, 2, and 3 are listed. Tunnels can be either initiator or responder role.
2. To perform a concise (short form) listing of of the specified phase 2 tunnels in the database with each preceded by the associated phase 1 tunnel, enter:

   ```
   ike cmd=list phase=2 numlist=1-3 db
   ```

   These are tunnels defined in the database which may or may not be currently active in the tunnel manager. All tunnels in the database are used in the initiator role only.
3. To perform a verbose (long form) listing of a phase 1 tunnel followed by all of its associated phase 2 tunnels from the tunnel manager, enter:

   ```
   ike cmd=list phase=1+ role=r verbose
   ```

   Only tunnels which were activated in the responder role are listed. All available tunnel numbers are listed since no **numlist** was specified.

remove

Purpose

Deactivates specified phase 1 or phase 2 tunnel(s).

Syntax **ike cmd=remove** [**phase=1**|**2**] [**numlist=** *tunnel_num_list*] [**all**]

Description The **remove** subcommand requests the deactivation of phase 1 or phase 2 tunnel(s). Because phase 2 tunnels are associated with a phase 1 tunnel, if a phase 1 tunnel is deactivated, all phase 2 tunnels under the phase 1 tunnel are not refreshed when the phase 2 tunnel lifetime expires.

Flags

phase

Indicates the phase of the tunnel to be deactivated and must be specified. A phase value of 1 refers to a phase 1 tunnel and a phase value of 2 refers to a phase 2 tunnel.

numlist

Lists the tunnel numbers you would like to deactivate. The **,** (comma) and **-** (dash) characters can be used to delimit values and indicate ranges. For example:

```
ike cmd=remove numlist=1,3,5-7
```

When **numlist** is omitted, all tunnels are deactivated.

all Deactivates all active tunnels. This parameter does not work in conjunction with **numlist**.

Examples

1. To deactivate phase 1 tunnels numbered 1, 2, and 3, enter:

   ```
   ike cmd=remove phase=1 numlist=1-3
   ```

2. To deactivate all phase 1 and phase 2 tunnels, enter:

   ```
   ike cmd=remove all
   ```

3. To deactivate all phase 2 tunnels but keep all phase 1 tunnels active, enter:

   ```
   ike cmd=remove phase=2 all
   ```

4. To deactivate all phase 1 tunnels (corresponding phase 2 tunnels will not be refreshed), enter:

   ```
   ike cmd=remove phase=1 all
   ```

log

Purpose Read the ISAKMP daemon log level from **/etc/isamkpd.conf** and start logging at that level.

Syntax **ike cmd=log**

Note: If the log level or the output file name in **/etc/syslog.conf** are changed, the **refresh -s syslogd** command must also be run.

Description The **log** subcommand causes the ISAKMP daemon to read the log level from **/etc/isakmpd.conf**, and a filename from **/etc/syslog.conf**. The logging level specified is set and the log output, along with other **syslog** output, is placed in the file specified.

Note There are four valid logging levels for the ISAKMP daemon. They are **none**, **errors**, **events**, and **information**. **none** means no logging, **errors** means logging of only ISAKMP daemon errors will occur, **events** means errors and other ISAKMP daemon events will be logged, and **information** is the highest level of logging which is all inclusive.

Files

/usr/sbin/ike	Location of the **ike** admin commands.
/etc/isakmpd.conf	Configuration file for the **iksakmpd** daemon.
/etc/syslog.conf	Provides configuration information for the **syslogd** daemon.

Related Information

The **syslog** subroutine.

The **syslog.conf** file.

The **syslogd** daemon.

The **ikedb** command.

ikedb Command

Purpose

Retrieves, updates, deletes, imports, and exports information in the IKE database.

Syntax

ikedb -p[**F s**] [**-e** *entity-file*] [*XML-file*]

ikedb -g[**r**] [**-t** *type* [**-n** *name* | **-i** *ID* **-y** *ID-type*]]

ikedb -d -t *type* [**-n** *name* | **-i** *ID* **-y** *ID-type*]

ikedb -c[**F**] [**-l** *linux-file*] [**-k** *secrets-file*] [**-f** *XML-file*]

ikedb -x

ikedb -o

Description

The **ikedb** command allows the user to write to (**put**) or read from (**get**) the IKE database. The input and output format is an Extensible Markup Language (XML) file. The format of an XML file is specified by its Document Type Definition (DTD). The **ikedb** command allows the user to see the DTD that is used to validate the XML file when doing a put. While entity declarations can be added to the DTD using the **-e** flag, this is the only modification to the DTD that can be made.

Any external DOCTYPE declaration in the input XML file will be ignored and any internal DOCTYPE declaration might result in an error. The rules followed to parse the XML file using the DTD are specified in the XML standard. **/usr/samples/ipsec** has a sample of what a typical XML file that defines common tunnel scenarios looks like.

Flags

-p Performs a **put**, which writes to the database, based on the given *XML-file*.

-F Forces a **put**, even if a specified tunnel, protection, proposal, group, or pre-shared key would overwrite one that already exists in the database. The default is for such **put** attempts to fail.

-s Swaps the local and remote IDs of all tunnels. This facilitates importing a tunnel generated by a peer machine. This flag only affects tunnels. This option is illegal if the remote ID of any tunnel is a group.

-e *entity-file*
Specifies the name of the file containing the `<!ENTITY ...>` lines as defined by *entity-file*. These lines are added to the internal DTD and allow the user to include XML files in other XML files.

XML-file
Specifies the **XML-file** to be used and must be the last argument to be displayed in the command line. The *XML-file* determines whether the write is to a tunnel, protection, proposal, group, pre-shared key, or all of these. If no *XML-file* is specified, input is read from **stdin**. A - (hyphen) can also be used to specify **stdin**.

-g Performs a **get**, which displays what is stored in the IKE database. Output is sent to **stdout** and is in XML format, which is suitable for processing with **ikedb -p**.

-r Recursive. If this flag is specified for a phase 1 tunnel, information is also returned for all associated phase 2 tunnels and all protections and proposals associated with both sets of tunnels.

-t *type* Specifies the *type* of output requested. *Type* can have the value of any of the XML elements under AIX_VPN, such as **IKETunnel**, **IPSecProtection**, and so on. If omitted, the entire database is output.

-n *name*
Specifies the *name* of the requested object. *Name* can be the name of a proposal, protection, tunnel, or group, depending on the value of the **-t** flag. The **-n** flag is valid with all values specified by the **-t** flag, except **IKEPresharedKey**. If omitted, all objects of the specified *type* will be output.

-i *ID* Specifies the *ID* associated with a pre-shared key. The **-i** flag is only valid with the **IKEPresharedKey** value of the **-t** flag. If omitted, all objects of the specified *type* will be output. The **-i** flag must be used in conjunction with the **-y** flag.

-y *ID-type*
Specifies the *ID-type* defined by the **-i** flag. ID-type can be any of the legal types allowed in the XML file, such as **User_FQDN**, **IPV4_Address**, and so on. The **-y** flag must be used in conjunction with the **-i** flag.

-d Performs a **delete** on the specified item from the database. The flags are the same as for the **-g** flag, except that **-r** is not supported.

-c Performs a conversion from a Linux IPSec configuration file to an AIX IPSec configuration file in XML format. It requires as input one or two files from Linux, a configuration file, and possibly a secrets file with pre-shared keys.

-F Forces a **put**, even if a specified tunnel, protection, proposal, group, or pre-shared key would overwrite one that already exists in the database. The default is for such **put** attempts to fail. The **-F** flag has no effect if the **-f** flag is also used.

-l *linux-file*
Specifies the Linux configuration file as define by *linux-file*. If no file is specified, the system looks for the **ipsec.conf** file in the current directory.

-k *secrets-file*
Specifies the Linux pre-shared keys file as defined by the *secrets-file* parameter. If no file is specified, the system looks for the **ipsec.secrets** file in the current directory.

-f *XML-file*
Specifies the XML configuration file to which the Linux configuration files are converted. The default behavior is to do a **put** directly to the IKE database. If the filename given is a hyphen (-), the results are sent to **stdout**.

-x Performs an **expunge** on the database. This empties out the database.

-o Performs an **output** of the DTD that specifies all elements and attributes for an XML file that is used by the **ikedb** command. The DTD is sent to **stdout**.

Files

/usr/samples/ipsec Examples of an XML file that sets up various tunnel configurations.

Examples

1. To **put** definitions to the IKE database from an XML file that has been generated on a peer machine and overwrite any existing objects in the database with the same name, type:

```
ikedb -pFs peer_tunnel_conf.xml
```

`peer_tunnel_conf.xml` is the XML file generated on a peer machine.

2. To **get** the definition of the phase 1 tunnel named `tunnel_sys1_and_sys2` and all dependent phase 2 tunnels with respective proposals and protections, type:

```
ikedb -gr -t IKETunnel -n tunnel_sys1_and_sys2
```

3. To **delete** all preshared keys from the database, type:

```
ikedb -d -t IKEPresharedKey
```

imake Command

Purpose

C preprocessor interface to the **make** command.

Syntax

imake [**-D***Define*] [**-I***Directory*] [**-T***Template*] [**-f** *FileName*] [**-C** *FileName*] [**-s** *FileName*] [**-e**] [**-v**]

Description

The **imake** command generates **Makefiles** from a template, a set of cpp macro functions, and a per-directory input file called **Imakefile**. This command keeps machine dependencies (such as compiler options, alternate command names, and special **make** command rules) separate from the descriptions of the items to build.

imake invokes cpp with any **-I** or **-D** flags passed on the command line and passes to it the following three lines:

```
#define IMAKE_TEMPLATE "Imake.tmpl"
#define INCLUDE_MAKEFILE "Imakefile"
#include IMAKE_TEMPLATE
```

Override **Imake.tmpl** and **Imakefile** by using the **-T** and **-f** flags, respectively.

The IMAKE_TEMPLATE typically reads the following files:

- A machine-dependent parameters file in which the parameters are specified as cpp symbols
- A site-specific parameters file
- A file that defines variables
- A file containing cpp macro functions for generating **make** command rules
- The **Imakefile** (specified by INCLUDE_IMAKEFILE) in the current directory.

The **Imakefile** file uses the macro functions to indicate what targets to build and the **imake** command generates the appropriate rules.

Imake configuration files contain two types of variables, imake variables and make variables. The imake variables are interpreted by cpp when the **imake** command is run. By convention, they are not case-sensitive. The make variables are written into the **Makefile** for later interpretation by the **make** command. By convention, make variables are uppercase.

The rules file (usually named **Imake.rules** in the configuration directory) contains a variety of cpp macro functions that are configured according to the current platform. The **imake** command replaces any occurrences of the string ``@@'' with a newline (carriage return) to support macros that generate more than one line of make rules. For example, the macro:

```
#define   program_target(program, objlist)        @@\
program:  objlist                                 @@\
          $(CC)  -o  $@  objlist  $(LDFLAGS)
```

when called with `program_target(foo,foo1.o foo2.o)` will expand to:

```
foo:      foo1.o  foo2.o
          $(CC)  -o  $@  foo1.o  foo2.o  $(LDFLAGS)
```

On systems whose cpp reduces multiple tabs and spaces to a single space, the **imake** command attempts to put back any necessary tabs (the **make** command distinguishes between tabs and spaces). For this reason, precede all colons (:) in command lines by a backslash (\).

Use with AIXwindows

AIXwindows uses the **imake** command extensively for both full builds within the source tree and builds of external software. Two special variables, TOPDIR and CURDIR, are set to make referencing files using relative path names easier. For example, the following command is generated automatically to build the **Makefile** in the **lib/X** directory (relative to the top of the sources):

```
%  ../../../config/imake  -I../../../config  \
             -DTOPDIR=../../.  -DCURDIR=./lib/X
```

To build AIXwindows programs outside the source tree, a special symbol, UseInstalled, is defined and the TOPDIR and CURDIR variables are omitted. If the configuration files are properly installed, you can use the **xmkmf** command.

The **imake** command reads the following files as used by AIXwindows.

Note: The indented format indicates files that include other files.

```
Imake.tmpl                  generic variables
   site.def                 site-specific, BeforeVendorCF defined
         *.cf               machine-specific
             *Lib.rules     shared library
   site.def                 site-specific, AfterVendorCF defined
   Imake.rules              rules
   Project.tmpl             X-specific variables
         *Lib.tmpl          shared library variables
Imakefile
   Library.tmpl             library rules
   Server.tmpl              server rules
   Threads.tmpl              multi-thread rules
```

Note: The **site.def** file is included twice, both before and after the ***.cf** file. Although most site customizations are specified after the ***.cf** file, some, such as the choice of compiler, need to be specified before, because other variable settings may depend on them.

The first time the **site.def** file is included, the **BeforeVendorCF** variable is defined, and the second time, the **AfterVendorCF** variable is defined. All code in the **site.def** file should be placed inside a **#ifdef** macro for one of these symbols.

Flags

-D*Define*	Passed directly to cpp to set directory-specific variables. For example, X-windows uses this flag to set the TOPDIR variable to the name of the directory containing the top of the core distribution, and the CURDIR variable to the name of the current directory, relative to the top.
-e	Indicates that the **imake** command should execute the generated **Makefile**. The default is to leave this to the user.
-f *FileName*	Specifies the name of the per-directory input file. The default is the **Imakefile** file.
-I*Directory*	(Uppercase i) Passed directly to cpp to indicate the directory in which the imake template and configuration files are located.
-C *FileName*	Specifies the name of the **.c** file that is constructed in the current directory. The default is **Imakefile.c**.
-s *FileName*	Specifies the name of the **make** description file to be generated, without invoking the **make** command. If the *FileName* variable is a **-** (dash), the output is written to **stdout**. The default is to generate, but not execute, a **Makefile**.

-T*Template* Specifies the name of the master template file (which is usually located in the directory specified with **-I**) used by the **cpp** command. The default is the **Imake.tmpl**.

-v Indicates that **imake** should print the **cpp** command line that it is using to generate the **Makefile**.

Environment Variables

Note: The following environment variables may be set, but their use is not recommended because they introduce dependencies that are not readily apparent when the **imake** command is run.

IMAKEINCLUDE If defined, specifies an include argument for the C preprocessor. For example:

```
-I/usr/include/local
```

IMAKECPP If defined, specifies a valid path to a preprocessor program. For example:

```
/usr/local/cpp
```

The default is the **/lib/cpp** program.

IMAKEMAKE Specifies a valid path to a **make** program such as **/usr/local/make**. By default, **imake** uses whatever **make** program is found using the **execvp** subroutine. This variable is only used if the **-e** flag is specified.

Example

```
imake -I/usr/lib/X11/config -DTOPDIR=/usr/lpp/X11/Xamples
```

Files

/usr/tmp/tmp-imake.*nnnnnn* Specifies the temporary input file for the cpp preprocessor.

/usr/tmp/tmp-make.*nnnnnn* Specifies the temporary input file for make.

/lib/cpp The default C preprocessor.

Related Information

The **make** command, **xmkmf** command.

---

imapd Daemon

Purpose

Starts the Internet Message Access Protocol (IMAP) server process. This command only applies to AIX 4.2.1 or later.

Syntax

imapd [**-c**]

Description

The **imapd** command is an IMAP4 server. It supports the IMAP4 remote mail access protocol. Also, it accepts commands on its standard input and responds on its standard output. You normally invoke the **imapd** command with the **inetd** daemon with those descriptors attached to a remote client connection.

The **imapd** command works with the existing mail infrastructure consisting of **sendmail** and **bellmail**.

Flags

-c Suppresses the reverse host name lookup.

Exit Status

All error and status information is written to a logfile if **syslogd** is configured for logging.

Security

Access Control: You must have root authority to run this command.

Auditing Events: N/A

Files

/usr/sbin/imapd	Contains the **imapd** command.
/etc/services	Specifies the file with port assignments for required services. The following entry must be in this file: `imap2  143/tcp  # Internet Mail Access Protocol`

Related Information

The **pop3d** daemon.

---

impfilt Command

Purpose

Imports filter rules from an export file.

Syntax

impfilt [**-v 4**|**6**] **-f** *directory* [**-l** *filt_id_list*]

Description

Use the **impfilt** command to import filter rules from text export file(s) that are generated by the **expfilt** command.

Flags

-v	IP version of the rules to be imported. The value of **4** specifies IP version 4 and the value of **6** specifies IP version 6. When this flag is not used, both IP version 4 and IP version 6 are imported.
-f	Specifies the directory where the imported text files are to be read.
-l	Lists the IDs of the filter rules to be imported. The filter rule IDs can be separated by ″**,**″. If this flag is not used, all filter rules for the applicable IP version(s) in the text export files will be imported.

Related Information

The **expfilt** command.

importvg Command

Purpose

Imports a new volume group definition from a set of physical volumes.

Syntax

importvg [**-V** *MajorNumber*] [**-y** *VolumeGroup*] [**-f**] [**-c**] [**-x**] **l** [**-L** *VolumeGroup*] [**-n**] [**-F**] [**-R**] [**-I**] *PhysicalVolume*

Description

Attention: When you issue the **importvg** command to a previously defined volume group, the QUORUM and AUTO ON values will be reset to volume group default values. You should verify the parameters of the newly imported volume group with the **lsvg** command and change any values with the **chvg** command.

The **importvg** command makes the previously exported volume group known to the system. The *PhysicalVolume* parameter specifies only one physical volume to identify the volume group; any remaining physical volumes (those belonging to the same volume group) are found by the **importvg** command and included in the import. An imported volume group is automatically varied unless the volume group is Concurrent Capable. You must use the **varyonvg** command to activate Concurrent Capable volume groups before you access them.

When a volume group with file systems is imported, the **/etc/filesystems** file is updated with values for the new logical volumes and mount points. After importing the volume group and activating it with the **varyonvg** command, you must run the **fsck** command before the file systems can be mounted. However, the mount point information would be missing from the LVCB (logical volume control block) if it is longer than 128 characters. In this case, the **importvg** command will not be able to update the **/etc/filesystems** file with the stanza for the newly imported logical volume. You should manually edit the **/etc/filesystems** file to add a new stanza for this logical volume.

The **importvg** command changes the name of a logical volume if the name already exists in the system. It prints a message and the new name to standard error, and updates the **/etc/filesystems** file to include the new logical volume name.

Notes:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. AIX Version 4 changed the behavior of **importvg** so that as part of the **importvg** process, the volume group is automatically varied on by the system after it is imported. However, if the volume group is Concurrent Capable then the **importvg** command prompts you to **varyonvg** the imported volume group manually.
3. A volume group with a mirrored striped logical volume cannot be back ported into a version older than AIX 4.3.3.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit importvg** fast path to run this command.

Flags

-c	This flag is ignored. On AIX 5.2 and higher only Enhanced Concurrent Capable volume groups will be created.
-f	Forces the volume group to be varied online.

-L *VolumeGroup* Takes a volume group and learns about possible changes performed to that volume group. Any new logical volumes created as a result of this command emulate the ownership, group identification, and permissions of the **/dev** special file for the volume group listed in the **-y** flag. The **-L** flag performs the functional equivalent of the **-F** and **-n** flags during execution.

Restrictions:

1. The volume group must not be in an active state on the system executing the **-L** flag.
2. The volume group's disks must be unlocked on all systems that have the volume group varied on and operational. Volume groups and their disks may be unlocked, remain active and used via the **varyonvg -b -u** command.
3. The physical volume name provided must be of a good and known state, the disk named may not be in the missing or removed state.
4. If an active node has both added AND deleted logical volumes on the volume group, the **-L** flag may produce inconsistent results. The **-L** flag should be used after each addition or deletion, rather than being deferred until after a sequence of changes.
5. If a logical volume name clash is detected, the command will fail. Unlike the basic **importvg** actions, clashing logical volume names will not be renamed.

-F Provides a fast version of **importvg** that checks the Volume Group Descriptor Areas of only the disks that are members of the same volume group. As a result, if a user exercises this flag, they must ensure that all physical volumes in the volume group are in a good and known state. If this flag is used on a volume group where a disk may be in missing or removed state, the command may fail or the results may be inconsistent.

-I Causes the **importvg** command to fail if imfs fails.

-n Causes the volume not to be varied at the completion of the volume group import into the system.

-R Restores the ownership, group ID, and permissions of the logical volume special device files. These values will be restored only if they were set using **U**, **G** and **P** flags of **mklv** and **chlv** commands. This flag is applicable only for big vg format volume groups only.

-V *MajorNumber* Specifies the major number of the imported volume group.

-x This flag is ignored. On AIX 5.2 and higher only Enhanced Concurrent Capable volume groups will be created.

Attention: This entry must be added after the entry used to initiate **srcmstr**.

-y *VolumeGroup* Specifies the name to use for the new volume group. If this flag is not used, the system automatically generates a new name.

The volume group name can only contain the following characters: ″A″ through ″Z,″ ″a″ through ″z,″ ″0″ through ″9,″ or ″_″ (the underscore), ″-″ (the minus sign), or ″.″ (the period). All other characters are considered invalid.

Examples

1. To import the volume group `bkvg` from physical volume `hdisk07`, enter:

   ```
   importvg  -y bkvg hdisk07
   ```

   The volume group `bkvg` is made known to the system.
2. To use the -L on a multi-tailed system:

   ```
   Node A has the volume group datavg varied on.
   Node B is aware of datavg, but it is not varied on.
   Node A: varyonvg -b -u datavg
   Node B: importvg -L datavg hdisk07
   Node A: varyonvg datavg
   ```

Files

/usr/sbin	Directory where the **importvg** command resides.
/tmp	Directory where the temporary files are stored while the command is running.

Related Information

The **exportvg** command, **varyonvg** command.

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the Logical Volume Manager, physical volumes, logical volumes, volume groups, organization, ensuring data integrity, and allocation characteristics.

For information on installing the Web-based System Manager, see Chapter 2: Installation and System Requirements in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The System Management Interface Tool in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the structure, main menus, and tasks that are done with SMIT.

---

imptun Command

Purpose

Adds the exported tunnel definitions and optional user-defined filter rules associated with the tunnels to the local host.

Syntax

imptun -f *directory* [**-t** *tunnel_id_list*] [**-v 4** | **6**] [**-n**] [**-r**] [**-g**] [**-l manual**]

Description

Use the **imptun** command to add exported tunnel definitions and optional user-defined filter rules associated with the exported tunnels (files generated by the tunnel owner by using the **exptun** command) to the local host. This command can also import tunnel definitions from the exported files generated by the IBM firewall (SNG) product export command.

A new tunnel ID is generated by the local host when a tunnel is imported to the local tunnel table. The auto-generated filter rules associated with the tunnel also is generated automatically. Importing the exported user-defined filter rules is optional.

If the exported files are transmitted by diskette, it is assumed they will be loaded to a local file directory using a command such as **tar**, depending on the tunnel owner's instructions.

Flags

-f	Specifies the directory from where the exported files will be read.
-g	The suppress system auto-generated filter rule flag. If the **-g** flag is not used, the **imptun** command generates two filter rules for each imported tunnel automatically. The auto-generated filter rules allow all traffic between the two end points of the tunnel to go through the tunnel. If the **-g** flag is specified, the command only imports the tunnel ibm definitions, and the user must add user-defined filter rules to use the tunnel.

-l	Specifies the type of the tunnel(s) you want to import. If **manual** is specified, only manual tunnel(s) are imported. **-n** and **-l** flags are mutually exclusive.
-n	Specifies that the export files were generated by the IBM firewall (version 2.2) tunnel export command. This flag cannot be specified with the **-v** flag. The **-n** flag is also mutually exclusive with the **-r** flag.
-r	Imports the user-defined filter rules associated with the tunnels that are being imported. To use the **-r** flag, it must have been specified with the **exptun** command when the exported files were generated. The **-r** flag is mutually exclusive with the **-n** flag.
-t	Lists the set of tunnel IDs to be imported from the export files. The tunnel definitions identified by these tunnel IDs are added to the local host. If this flag is not used, all the tunnel definitions in the export files are added to the local host.
-v	Specifies the IP version of the tunnel definitions from the exported files that you wish to import. If the **-v** flag is not given, then all IP version 4 and IP version 6 tunnel definitions that exist in the export files are imported.

Related Information

The **gentun** command, **chtun** command, **rmtun** command, **exptun** command, **mktun** command, and **lstun** command.

---

inc Command

Purpose

Files new mail in a folder.

Syntax

inc [**+** *Folder*] [**-noaudit** | **-audit** *File*] [**-changecur** | **-nochangecur**] [**-form** *FormFile* | **-format** *String*] [**-help**] [**-file** *File*] [**-truncate** | **-notruncate**] [**-nosilent** | **-silent**] [**-width** *Number*]

Description

The **inc** command files incoming mail in a specified folder and outputs a list of the messages filed. A folder is a system directory. By default, the **inc** command removes the new messages from your mail drop and places them in the specified folder. To file new mail without deleting the mail drop, use the **-notruncate** flag.

If the specified folder does not exist, the **inc** command prompts you for permission to create it. The system creates the folder as a subdirectory of the user's Message Handler (MH) directory. The default folder is **inbox**.

> **Note:** If you do not have a `Path:` entry specified in your **.mh_profile** file, the **inc** command creates the folder as a subdirectory of the current directory.

Filed messages are assigned consecutive message numbers starting with the next highest number in the folder. Each new message receives the protection code specified in the `Msg-Protect:` entry in your **.mh_profile** file. If the `Msg-Protect:` entry does not exist, a protection code of 644 is assigned. If the `Unseen-Sequence:` entry exists, new messages are added to each sequence specified by the entry.

Flags

-audit *File*	Copies the current date to the specified file and appends the output of the **inc** command to the file.
-changecur	Sets the first new message as the current message for the specified folder. This flag is the default.
-file *File*	Files messages from the specified file instead of the user's maildrop.
+*Folder*	Specifies the folder in which to place new messages. By default, the system creates a subdirectory called **inbox** in the user's MH directory.
-form *FormFile*	Identifies a file that contains an alternate output format for the **inc** command.
-format *String*	Specifies a string that defines an alternate output format for the **inc** command.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-noaudit	Suppresses recording of information about any new messages filed. This is the default.
-nochangecur	Prevents alteration of the current message for the specified folder.
-nosilent	Prompts the user for any necessary information. This flag is the default.
-notruncate	Prevents clearing of the mailbox or file from which the **inc** command is taking new messages. If the **-file** flag is specified, the **-notruncate** flag is the default.
-silent	Prevents prompting by the **inc** command for information. This flag is useful when running the **inc** command in the background.
-truncate	Clears the mailbox or file from which the **inc** command is taking new messages. If the **-file** flag is not specified, the **-truncate** flag is the default.
-width *Number*	Sets the number of columns in the 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 alternate mailboxes.
`Folder-Protect:`	Sets the protection level for new folder directories.
`Msg-Protect:`	Sets the protection level for new message files.
`Path:`	Specifies the user's MH directory.
`Unseen-Sequence:`	Specifies the sequences used to keep track of unseen messages.

Examples

1. To incorporate new mail into the default mail folder, **inbox**, enter:

   ```
   inc
   ```

   If the **inbox** folder exists, the system displays a message similar to the following:

   ```
   Incorporating new mail into inbox...
   65+        04/08    jim@athena.a         Meeting                              <<The meeting will
   66         04/08    jim@athena.a         Schedule                             <<Schedule change
   ```

   In this example, two messages are filed in the `inbox` folder. The subject of the first message is `Meeting`, and the first line starts with the words `The meeting will`. The subject of the second message is `Schedule`, and the first line starts with the words `Schedule change`.

2. To incorporate new mail into a new folder called **testcases**, enter:

   ```
   inc     +testcases
   ```

   The system prompts you as follows:

   ```
   Create folder "/home/mary/testcases"?
   ```

If you wish to create the folder, enter:

```
yes
```

A message similar to the following is displayed:

```
Incorporating new mail into testcases...
67+        04/08    jim@athena.a            Meeting                                  <<We will begin
68         04/08    jim@athena.a            Schedule                                 <<Schedule change
```

Files

$HOME/.mh_profile	Customizes the MH user profile.
/etc/mh/mtstailor	Tailors the MH environment to the local environment.
/var/spool/mail/$USER	Specifies the location of the mail drop.
/usr/bin/inc	Contains the **inc** command.

Related Information

The **mhmail** command, **post** command, **scan** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

indent Command

Purpose

Reformats a C language program.

Syntax

indent *InputFile* [*OutputFile*] [**-nbad** | **-bad**] [**-nbap** | **-bap**] [**-nbbb** | **-bbb**] [**-nbc** | **-bc**] [**-br** | **-bl**] [**-c***n*] [**-cd***n*] [**-ncdb** | **-cdb**] [**-nce** | **-ce**] [**-ci***n*] [**-cli***n*] [**-d***n*] [**-di***n*] [**-ndj** | **-dj**] [**-nei** | **-ei**] [**-fa**] [**-nfa**] [**-nfc1** | **-fc1**] [**-i***n*] [**-nip** | **-ip**] [**-l***n*] [**-lc***n*] [**-nlp** | **-lp**] [**-npro**] [**-npcs** | **-pcs**] [**-nps** | **-ps**] [**-npsl** | **-psl**] [**-nsc** | **-sc**] [**-nsob** | **-sob**] [**-nslb** | **-slb**] [**-st**] [**-troff**] [**-nv** | **-v**] [**-T***Type*] ...

Description

The **indent** command reformats a C program as specified by flags entered with the command.

If you only specify the *InputFile* parameter, the reformatted file is written back into the *InputFile* parameter and a backup copy of the *InputFile* parameter is written in the current directory with a **.BAK** filename suffix.

If you specify the *OutputFile* parameter, the **indent** command checks to make sure its name is different from the *InputFile* parameter.

To set up your own profile of defaults for the **indent** command, create a file called **.indent.pro** in your login directory or the current directory. In this file, include as many flags as desired, separated by spaces, tabs, or new lines.

Flags in the **.indent.pro** file in the current directory override those in your login directory (with the exception of **-T***Type* flags, which accumulate). If the **indent** command is run and a profile file exists, the profile file is read to set up the defaults of the program. Flags on the command line, however, override profile flags.

Comment Handling
The **indent** command assumes that any comment with a **-** (dash) or ***** (asterisk) immediately after the start of a comment marker (**/*-** or **/****) is a comment surrounded by asterisks. Each line of the comment is left unchanged, except for its indentation. This indentation can be adjusted to account for the change in indentation of the first line of the comment.

All other comments are treated as text. The **indent** command fits as many words (separated by blanks, tabs, or new-lines) on a line as possible. Blank lines break paragraphs.

A block comment is a comment that is not to the right of the code, and extends for more than one line.

If a comment is on a line with code, it is started in the comment column set by the **-c***n* flag. Otherwise, the comment is started at *n* indentation levels less than where code is currently being placed, where *n* is specified by the **-d***n* flag. If the code on a line extends past the comment column, the comment starts further to the right. The right margin can be extended automatically in extreme cases.

Preprocessor Lines Handling
In general, the **indent** command leaves preprocessor lines alone. The only reformatting it does is to straighten up trailing comments. It leaves embedded comments alone. Conditional compilation (code between **#ifdef** and **#endif** lines) is recognized and the **indent** command attempts to compensate correctly for the syntactic peculiarities introduced.

C Syntax Handling
The parser built into the **indent** command attempts to cope with incomplete and misformed syntax. In particular, the use of macros like:

```
#define forever for(;;)
```

is handled properly. For best results, use the **indent** command on source that is syntactically correct.

Flags
Note: Flags can appear before or after file names.

-bad Forces a blank line after every block of declarations.

-nbad Suppresses a blank line after every block of declarations; active unless turned off with the **-bad** flag.

-bap Forces a blank line after every procedure body.

-nbap Suppresses a blank line after every procedure body; active unless turned off with the **-bap** flag.

-bbb Forces a blank line before every block comment.

-nbbb Suppresses a blank line before every block comment; active unless turned off with the **-bbb** flag.

-bc Forces a new line after each comma in a declaration.

-nbc Suppresses a new line after each comma in a declaration; active unless turned off with the **-bc** flag.

-bl Formats compound statements, structure initializations, and enum initializations, as follows:

```
if (...)
{
   code
}
```

-br Formats compound statements, structure initializations, and enum initializations, as follows:

```
 if (...) {
    code
}
```

This flag is active unless turned off with the **-bl** flag.

-c*n* Sets the initial tab position for comments on code to the *n* variable. The default value is 33.

-cd*n* Sets the initial tab position for comments on declarations to the *n* variable. By default, this flag uses the value defined with the **-c** flag.

-cdb Enables placing comment delimiters on blank lines; active unless turned off with the **-ncdb** flag. The **-cdb** flag affects only block comments, not comments to the right of code. Resulting comments look like the following:

```
/*
 * this is a comment
 */
```

-ncdb Disables placing comment delimiters on blank lines. The **-ncdb** flag affects only block comments, not comments to the right of code. Resulting comments look like the following:

```
/* this is a comment */
```

-ce Enables forcing **else** statements to follow the immediately preceding **}** (left bracket); active unless turned off with the **-nce** flag.

-nce Disables forcing **else** statements to follow the immediately preceding **}** (left bracket).

-ci*n* Indents the continuation lines *n* positions from the beginning of the first line of the statement. Expressions in parentheses have extra indentation added to indicate the nesting, unless the **-lp** flag is in effect. By default, this flag uses the value defined by the **-i** flag.

-cli*n* Indents the case labels *n* positions to the right of the containing flag statement. Entering **-cli0.5** causes case labels to be indented half a tab stop. This option is the only one that takes a fractional argument. By default, the value is **-cli0**.

-d*n* Controls the placement of comments that are not to the right of code with the *n* variable. Specifying the **-d1** flag causes such comments to appear one indention level to the left of code. By default, this flag uses **-d0** and comments are aligned with code. The location of comment lines relative to program code affects the comment indention.

-di*n* Specifies the number of positions to indent an identifier from a preceding declaration keyword with the *n* variable. By default, this flag uses **-di16**.

-dj Left-justifies declarations.

-ndj Indents declarations; active unless turned off with the **-dj** flag.

-ei Enables special **else**-**if** processing; active unless turned off with the **-nei** flag. The **-ei** flag causes **if** statements following **else** statements to have the same indentation as the preceding **if** statement.

-nei Disables special **else**-**if** processing.

-fa Flips assign operators from old style C code to the ANSI format. This flag remains active unless turned off with the **-nfa** flag.

> **Attention:** The possibility of changing the meaning of the code exists if the code was meant for the ANSI compiler. For example, `A=-B` becomes `A-=B`.
>
> **Note:** Use no spaces between operators. If the user means subtraction, then the flipping is necessary; on the other hand, if the user means A equals the negative of B, the flipping alters the meaning.

-nfa Suppresses flipping the operators. Use this flag if the code is written for an ANSI compiler.

-fc1 Enables formatting comments that start in column 1; active unless turned off with the **-nfc1** flag.

-nfc1 Disables formatting comments that start in column 1.

-i*n* Sets the indentation level size. By default, the level size is 8 positions.

-ip Enables indenting parameter declarations; active unless turned off with the **-nip** flag.

-nip Disables indenting parameter declarations.

-l*n* Sets the maximum column position of comments that are to the right of the code. If the comment does not fit on a line, a maximum of 25 characters are printed.

-lc*n* Sets the maximum line length for block comments to the *n* variable. By default, this flag uses the length specified with the **-l** flag.

-lp Aligns code surrounded by parentheses in continuation lines; active unless turned off with the **-nlp** flag. If a line has a left parenthesis with no matching right parenthesis on that line, continuation lines start at the position following the left parenthesis.

With the **-lp** flag in effect, such lines appear as follows:

```
p1 = first_procedure(second_procedure(p2,p3),
                     third_procedure(p4,p5));
```

Inserting two more new lines yields the following:

```
p1 = first_procedure(second_procedure(p2,
                                      p3),
                     third_procedure(p4,
                                     p5));
```

-nlp Leaves code surrounded by parentheses in continuation lines unaligned. With the **-nlp** flag in effect, such lines appear as follows:

```
p1 = first_procedure(second_procedure(p2,p3),
         third_procedure(p4, p5));
```

-npro Causes the profile files **./.indent.pro** and **$HOME/.indent.pro** to be ignored.

-pcs Inserts a space between each procedure call name and the following **(** (left parenthesis).

-npcs Suppresses a space between each procedure call name and the following **(** (left parenthesis); active unless turned off with the **-pcs** flag.

-ps Inserts spaces on both sides of the pointer following the **->** operator.

-nps Suppresses spaces on both sides of the pointer following the **->** operator; active unless turned off with the **-ps** flag.

-psl Left-justifies the names of procedures being defined; active unless turned off with the **-npsl** flag. The procedure types, if any, remain on the previous lines.

-npsl Disables left-justification of names of defined procedures.

-sc Enables the placement of ***** (asterisks) to the left of comments; active unless turned off with the **-nsc** flag.

-nsc Disables the placement of ***** (asterisks) to the left of comments.

-slb Treats any single-line comment that is not to the right of the code as a block comment.

-nslb Disables treating any single-line comment that is not to the right of the code as a block comment; active unless turned off with the **-slb** flag.

-sob Removes optional blank lines. Works in combination with any of the following flags: **-nbad**, **-nbap**, or **-nbbb**. Removes only blank lines that were inserted by the **-bad**, **-bap**, or **-bbb** flags.

-nsob Retains optional blank lines; active unless turned off with the **-sob** flag.

-st Causes the **indent** command to take its input from stdin and output to stdout.

-T*Type* Adds the *Type* variable to the list of type keywords. Names accumulate so **-T** can be specified more than once. You should specify all the types appearing in your program defined by **typedef** statements to produce the best output from the **indent** command.

-troff Formats the C program for processing by **troff**. Produces a listing similar to listings produced by the **vgrind** command. If no output file is specified, the default is standard output, rather than formatting in place.

-v Turns on verbose mode, which reports when one line of input is split into two or more lines of output and gives size statistics at completion.

-nv Turns off verbose mode; active unless turned off with the **-v** flag.

Examples

1. To format the `test.c` file using the default settings of the **indent** command and place the output into the `newtest.c` file, enter:

   ```
   indent test.c newtest.c
   ```

2. To format the `test.c` file so that a blank line is forced after every block of declarations and procedure body, use all other default settings, and store the output in the `newtest.c` file, enter:

   ```
   indent test.c newtest.c -bad -bap
   ```

3. To format the `test.c` file using the default settings of the **indent** command and to define `uint` as a type keyword recognizable to the **indent** command, enter:

```
indent test.c newtest.c -Tuint
```

Files

./.indent.pro	Contains the profile file.
$HOME/.indent.pro	Contains the profile file.
/usr/ccs/bin/indent	Contains the **indent** command.

Related Information

The **cb** command.

Commands Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

indxbib Command

Purpose

Builds an inverted index for a bibliography.

Syntax

indxbib *Database* ...

Description

The **indxbib** command makes an inverted index to the named database (or files) for use by the **lookbib** and **refer** commands. These files contain bibliographic references (or other kinds of information) separated by blank lines.

> **Note:** The **indxbib** command expects the database to exist in the current working directory.

A bibliographic reference is a set of lines, constituting fields of bibliographic information. Each field starts on a line beginning with a **%** (percent sign), followed by a key letter, then a space character, and finally the contents of the field, which can continue until the next line starting with a **%** (percent sign). All key letters are ASCII characters.

The **indxbib** command is a shell script that calls the **/usr/lib/refer/mkey** and **/usr/lib/refer/inv** files. The first program, **mkey**, performs the following operations:

1. Truncates words (delimited by blanks or tabs) to six characters.
2. Maps uppercase to lowercase characters.
3. Discards words shorter than three characters.
4. Discards the most commonly used words according to an existing **ign** file. An English language file, **/usr/lib/eign**, has been provided with a list of common English words. It is suggested, but not necessary, that users create their own files, named **ign**, consisting of language-specific common words. This file, if created, should exist in the **/usr/lib/nls/msg/$LANG** directory.
5. Discards numbers (dates) less than 1900 or greater than 2099.

   > **Note:** All dates should be indexed because many disciplines refer to literature written in the 1800s or earlier.

These parameters can be changed. (For more information, see Lesk, M.E. *Some Applications of Inverted Indexes on the* UNIX *System*.)

The second program, **inv**, creates in the working directory an entry file (**.ia**), a posting file (**.ib**), and a tag file (**.ic**).

Files

/usr/lib/eign	Contains the default list of common words the **indxbib** command discards while processing.
Database.**ia**	Contains the entry file.
Database.**ib**	Contains the posting file.
Database.**ic**	Contains the tag file.

Environment Variables

NLSPATH Refers to a list of directory names where the message catalog files can be found.

Related Information

The **addbib** command, **lookbib** command, **refer** command, **roffbib** command, **sortbib** command.

---

inetd Daemon

Purpose

Provides Internet service management for a network.

Syntax

Note: Use SRC commands to control the **inetd** daemon from the command line. Use the **rc.tcpip** file to start the daemon with each system restart.

/usr/sbin/inetd [**-d**] [**-t** *SecondsToWait*] [*ConfigurationFile*]

Description

The **/usr/sbin/inetd** daemon provides Internet service management for a network. This daemon reduces system load by invoking other daemons only when they are needed and by providing several simple Internet services internally without invoking other daemons.

The **inetd** daemon starts by default each time you start your system. When the daemon starts, it reads its configuration information from the file specified in the *ConfigurationFile* parameter. If the parameter is not specified, the **inetd** daemon reads its configuration information from the /**etc**/**inetd.conf** file.

Once started, the **inetd** daemon listens for connections on certain Internet sockets in the /**etc**/**inetd.conf**. The /**etc**/**inetd.conf** file describes to the **inetd** daemon how Internet service requests on Internet sockets should be handled. When the **inetd** daemon receives a request on one of these sockets, it determines which service corresponds to that socket and then either handles the service request itself or invokes the appropriate server.

Subservers of the inetd Daemon

The **inetd** daemon (a subsystem) controls the following daemons (subservers):

- **comsat** daemon
- **ftpd** daemon
- **fingerd** daemon
- **rlogind** daemon
- **rexecd** daemon

- **rshd** daemon
- **talkd** daemon
- **telnetd** daemon
- **tftpd** daemon
- **uucpd** daemon.

The **ftpd**, **rlogind**, **rexecd**, **rshd**, **talkd**, **telnetd**, and **uucpd** daemons are started by default. The **tftpd**, **fingerd**, and **comsat** daemons are not started by default unless they are uncommented in the **/etc/inetd.conf** file.

Inetd Configuration File

The **/etc/inetd.conf** file can be updated by using the System Management Interface Tool (SMIT), the System Resource Controller (SRC), or by editing the **/etc/inetd.conf**.

If you change the **/etc/inetd.conf**, using SMIT, then the **inetd** daemon will be refreshed automatically and will read the new **/etc/inetd.conf** file. If you change the file using your favorite editor, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file.

The entries in the /etc/inetd.conf file include the following information:

Service Name	Specifies the name of a valid Internet service.
Socket Type	Specifies the type of Internet socket used for the Internet service. (Only stream and datagram sockets are implemented.) Valid values are: stream dgram sunrpc_udp sunrpc_tcp
Protocol	Specifies the Internet Protocol used for the Internet service. Valid values are: tcp tcp6 udp udp6
Wait/Nowait	Specifies whether the **inetd** daemon should wait for the service to complete before continuing to listen for this type of service request.
Wait/Nowait	Specifies whether the **inetd** daemon should wait for the service to complete before continuing to listen for this type of service request. SRC works like wait, but instead of forking and waiting for the child to die, it does a **startsrc** on the subsystem and store information about the starting of the service. When the service is removed from the **inetd.conf** file and **inetd** is restarted, the service has a **stopsrc** issued to the service to stop it.
User	Specifies the user name that **inetd** should use to start the subserver.
Path	Specifies the fully qualified path name that inetd should execute to provide the service. For services that **inetd** provides internally, this entry should be internal.
Command	Specifies the name of the service to start and its parameters. This field is empty for internal services.

The **inetd** daemon can be run with or without the SRC. In addition, the inetd daemon can be controlled by issuing signals using the kill command.

Flags

-d	Sends debugging messages to the **syslogd** daemon.
-t *SecondsToWait*	Specifies the number of seconds to wait in the select() system call before looping. The *SecondsToWait* can be a number from 1 to 999999. Without this flag the **inetd** daemon will block until one of the active services is requested by a network connection. This flag should only be used when a machine is servicing many wait services like **tftp** and is not being used for other services. Since timing out the select() system call will cause the **inetd** daemon to use more CPU cycles, this flag is not recomended for most situations.

Service Requests

The Internet service requests that are supported internally by the **inetd** daemon are generally used for debugging. They include the following internal services:

ECHO	Returns data packets to a client host.
DISCARD	Discards received data packets.
CHARGEN	Discards received data packets and sends predefined or random data.
DAYTIME	Sends the current date and time in user-readable form.
TIME	Sends the current date and time in machine-readable form.

Related Information

The **fingerd** daemon, **ftpd** daemon, **rexecd** daemon, **rlogind** daemon, **rshd** daemon, **syslogd** daemon, **talkd** daemon, **telnetd** daemon, **tftpd** daemon.

The **inetd.conf** file format, **protocols** file format, **services** file format.

TCP/IP Daemons in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

infocmp Command

Purpose

Manages **terminfo** descriptions.

Syntax

infocmp [**-d**] [**-c**] [**-n**] [**-I**] [**-L**] [**-C**] [**-r**] [**-u**] [**-s** { **d**| **i**| **l**| **c**}] [**-v**] [**-V**] [**-1**] [**-w** *Width*] [**-A** *Directory*] [**-B** *Directory*] [*TermName...*]

Description

The **infocmp** command manages **terminfo** descriptions. You can use this command to:

- Compare a binary **terminfo** entry with other **terminfo** entries.
- Print a **terminfo** description from the binary file.
- Rewrite a **terminfo** description to take advantage of the **use** attribute.

The **infocmp** command prints the Boolean attributes first, the numeric attributes second, and the string attributes last.

Comparing Entries

Use the **-d**, **-c**, and **-n** flags to compare entries. The **-d** flag returns the differences between entries. The **-c** flag produces a list of the capabilities that are set and in common between two entries. The **-n** flag returns a list of the capabilities that neither entry has.

To compare **terminfo** entries, you specify two or more *TermName* parameters. The **infocmp** command compares the **terminfo** description of the first *TermName* parameter with each of the descriptions for the subsequent *TermName*s specified. If a capability is defined for only one of the terminal descriptions, the value returned will depend on the type of capability. For Boolean capabilities the **infocmp** command returns an F, the command returns a -1 for integer capabilities, and null for string capabilities.

Producing a Source Listing

Use the **-I** (uppercase i), **-L**, **-C**, and **-r** flags to produce a source listing for one or more terminals. If you do not specify a *TermName* parameter, the system uses the **TERM** environment variable. You can use these source options to produce a source file for a terminfo binary when one is not available.

The **I** (uppercase i) flag produces a listing with the terminfo names. The **-L** flag produces a listing using the long **C** variable names listed in **/usr/include/term.h**.

The **-C** flag uses **termcap** names instead of terminfo capability names when producing the source listing. The **infocmp** commands translates and outputs only those **terminfo** capabilities that have a corresponding **termcap** code name. To remove this restriction, specifying the **-r** flag. This flag causes the command to output **terminfo** capabilities that cannot be translated into **termcap** format.

When using the **-C** and **-r** flags, the **infocmp** command notes any string parameters it was unable to convert to the **termcap** format. You must edit these parameters manually. The command collects all padding information for strings together and places it at the beginning of the string where **termcap** expects it. Mandatory padding is optional after translation. Mandatory padding is padding information with a trailing **/** (slash).

> **Note:** The **-C** and **-r** flags cannot always convert a **terminfo** string into its equivalent **termcap** form. Similarly, a conversion from the **termcap** file format back into the **terminfo** file format does not necessarily reproduce the original source.

Definitions with the use Attribute

Given a list of terminal menus and the **-u** flag, the **infocmp** command compares the first terminal's description against the other terminal descriptions. The **infocmp** command then creates a new description for the first terminal using as much of the subsequent terminal descriptions as possible.

When you specify the **-u** flag and a list of terminal names, the **infocmp** command does the following:

- Compares subsequent terminal descriptions against the first.
- Creates a description of the first terminal you specified relative to the description of the other terminals.

The new description for the first terminal will have the following:

- Capabilities that exist in the subsequent terminals but do not exist for the first terminal will appear with an @ in the resulting description.

  > **Note:** The @ implies that the capability does not exist.
- Capabilities defined in a subsequent terminal with the same value are replaced with *use=<subsequent terminal>*.
- Any capabilities in the first terminal not found in any of the other terminals are printed along with the corresponding values.
- If the first terminal has a capability whose value differs from the value found in at least one of the other terminals, the capability is printed.

You can change a description and specify a capability after the **use** attribute. If this capability is also found in the terminal referenced by the **use** attribute, the second capability takes precedence over the one referenced by the **use** attribute.

Changing Databases

By default, terminal descriptions appear in the system **terminfo** database directory, **/usr/share/lib/terminfo**. You can specify a different database location with the **TERMINFO** environment variable. The **infocmp** command first checks to see if this variable exists. If the variable does not exist, the command uses the system **terminfo** database.

You can use the **-A** and **-B** flag with the **infocmp** command to override the system database. The **-A** flag identifies the **terminfo** database for the first *TermName* parameter. The **-B** flag identifies the database to use for any subsequent terminals you name. Together, these flags make it possible to compare descriptions for two terminals with the same name located in two different databases.

Flags

-A *Directory*
: Identifies the **terminfo** database for the first *TermName* parameter.

-B *Directory*
: Identifies the **terminfo** database for every *TermName* parameter except the first.

-C
: Uses the **termcap** code names to produce the source listing. Will not list **terminfo** capabilities that cannot be translated to **termcap** format.

-c
: Lists the capabilities that are common between the two entries. Capabilities that are not set are ignored. This flag can be used as a quick check to see if it is desirable to use the **-u** flag.

-d
: Lists the capabilities that are different between terminals. You can use this flag to pinpoint the differences between similar terminal entries.

-I (uppercase i)
: Uses the **terminfo** capability names when producing the source listing.

-1 (numeral)
: Prints the capabilities one to a line. by default, the fields are printed several to a line to a maximum width of 60 characters.

-L
: Uses the long C variable name listed in **/usr/include/term.h** file to produce the source listing.

-n
: Compares two entries and lists the capabilities that do not exist in either. If you do not specify a *TermName* parameter, the system uses the **TERM** environment variable for both *TermName* parameters. You can use this as a quick check to see if anything was left out of the description.

-r
: Instructs the **infocmp** command to output **terminfo** capabilities that cannot be translated to **termcap** format. This flag is valid only with the **-C** flag.

-s
: Sorts the output from the **infocmp** command within each capability type (Boolean, numeric, and string) and according to the argument below:

  d Sort in the order specified in the **terminfo** database.

  i Sort by **terminfo** name.

  l Sort by the long C variable name.

  c Sort by the **termcap** name.

  If you do not specify an option with the **-s** flag, the command sorts each capability alphabetically by the **terminfo** name within each type. If you specify the **-C** or the **-L** flags with the **-s** flag, the capabilities are sorted by the **termcap** name or the long C variable name, respectively.

-u
: Compares two or more terminal descriptions and produces new descriptions using the **use** attribute.

-v
: Prints out tracing information on standard error.

-V
: Prints out the version of the program in use on standard error and exits.

-w *Width*
: Changes the output to the specified number of characters per line. The output includes as many fields as possible that can fit within the specified number of characters.

  Note: Fields are not truncated.

Examples

1. To list the common capabilities between the aixterm and lft terminals, enter:
   ```
   infocmp -c aixterm lft
   ```
2. To list all of the capabilities that are possible but do not currently exist for the current terminal, enter:
   ```
   infocmp -n
   ```
3. To produce a source listing for the lft terminal in **terminfo** format, enter:
   ```
   infocmp -I lft
   ```
4. To produce a source listing for the terminal description my_term that is located in **/tmp** using as much of the lft description as possible, enter:
   ```
   infocmp -A /tmp -u my_term lft
   ```

File

/usr/share/lib/terminfo	Contains the compiled terminal description database.

Related Information

The **tic** and **captoinfo** commands.

The **terminfo** file format.

---

infocenter Command

Purpose

Starts the IBM eServer pSeries Information Center in a browser window.

Syntax

infocenter

Description

The **infocenter** command starts the IBM eServer pSeries Information Center in a browser window. The Information Center is the central point from which AIX and IBM eServer pSeries documentation, AIX FAQs, the Message Center, and How-to documentation can be accessed. The Information Center can also be started by clicking the **Information Center** icon on the **Help** panel available from the CDE toolbar.

Note: The Information Center is available only by means of an HTML browser that has access to the Internet. The **infocenter** command (and the **Information Center** icon) activates the browser and directs the browser to the Web address of the Information Center.

The Web address of the Information Center is in the **/usr/lpp/bosinst/bos.vendor.profile** file. If the Information Center Web address is provided in this file, it will be in the following format:

```
export INFORMATION_CENTER_URL=information center url
```

If a line similar to this does not exist in the **/usr/lpp/bosinst/bos.vendor.profile** file, or if the file does not exist, the user will receive a message indicating that the Information Center cannot start because an Information Center URL was not found.

Files

/usr/sbin/infocenter	Starts the Information Center in a browser window.

/usr/lpp/bosinst/bos.vendor.profile Contains vendor profile information including the Web address used by the **infocenter** command.

install Command

Purpose

Installs a command.

Syntax

/usr/bin/install [**- c** *DirectoryA*] [**- f** *DirectoryB*] [**- i**] [**- m**] [**- M** *Mode*] [**- O** *Owner*] [**- G** *Group*] [**- S**] [**- n** *DirectoryC*] [**- o**] [**- s**] *File* [*Directory ...*]

Description

The **install** command installs a specified file in a specific place within a file system. It is most often used in makefiles. When replacing files, the **install** command copies (or moves) each file into the appropriate directory, thereby retaining the original owner and permissions based on the behavior of the **cp** and **mv** commands. An attempt is made to change the destination to owner **bin** and group **bin**. The **-O** *Owner* and **-G** *Group* flags can be used to specify a different owner or group. The **install** command writes a message telling you exactly which files it is replacing or creating and where they are going.

You must be a super-user if you want to specify the ownership of the installed file with the **-O** or **-G** flags.

If you do not specify the *Directory* parameter, the **install** command searches a set of default directories (**/usr/bin**, **/etc**, and **/usr/lib**, in that order) for a file with the same name as the *File* parameter. The first time it finds one, it overwrites it with *File* and issues a message indicating that it has done so. If a match is not found, the **install** command issues a message telling you there was no match and exits with no further action. If the *File* parameter does not exist in the current directory, the **install** command displays an error message and exits with a nonzero value.

If any directories are specified on the command line, the **install** command searches them before it searches the default directories.

Flags

-c *DirectoryA*	Installs a new command file in the *DirectoryA* variable only if that file does not already exist there. If it finds a copy of *File* there, it issues a message and exits without overwriting the file. This flag can be used alone or with the **-s**, **-M**, **-O**, **-G**, or **-S** flag.
-f *DirectoryB*	Forces installation of *File* in *DirectoryB* whether or not *File* already exists. If the file being installed does not already exist, the command sets the permission code and owner of the new file to **755** and **bin**, respectively. This flag can be used alone or with the **-o**,**-s, -M, -O, -G,** or **-S** flag.
-G *Group*	Specifies a different group for the destination file. The default group is **bin**.
-i	Ignores the default directory list and searches only those directories specified on the command line. This flag cannot be used with the **-c, -f,** or **-m** flags.
-m	Moves the *File* parameter to the directory instead of being copied. Cannot be used with the **-c**, **-f**, **-i**, or **-n** flag.
-M *Mode*	Specifies the mode of the destination file.
-n *DirectoryC*	Installs the *File* parameter in the *DirectoryC* variable if it is not in any of the searched directories, and sets the permissions and owner of the file to **755** and **bin**, respectively. This flag cannot be used with the **-c**, **-f**, or **-m** flag.
-o	Saves the old copy of the *File* parameter by copying it into a file called **OLD***File* in the same directory. This flag cannot be used with the **-c** flag.
-O *Owner*	Specifies a different owner of the destination file. The default owner is **bin**.

-s Suppresses the display of all but error messages.
-S Causes the binary to be stripped after installation.

Examples

1. To replace a command that already exists in one of the default directories, enter:

   ```
   install fixit
   ```

   This replaces the **fixit** file if it is found in the **/usr/bin**, **/etc**, or **/usr/lib** directory. Otherwise, the **fixit** file is not installed. For example, if **/usr/bin/fixit** exists, then this file is replaced by a copy of the file **fixit** in the current directory.
2. To replace a command that already exists in a specified or default directory and to preserve the old version, enter:

   ```
   install  -o fixit /etc /usr/games
   ```

   This replaces the **fixit** file if it is found in the **/etc** or **/usr/games** directory or in one of the default directories. Otherwise the **fixit** file is not installed. If the file is replaced, the old version is preserved by renaming it **OLDfixit** in the directory in which it was found.
3. To replace a command that already exists in a specified directory, enter:

   ```
   install  -i fixit /home/jim/bin /home/joan/bin /usr/games
   ```

   This replaces the **fixit** file if it is found in the **/home/jim/bin**, **/home/joan/bin**, or **/usr/games** directory. Otherwise, the file is not installed.
4. To replace a command found in a default directory or install it in a specified directory if it is not found, enter:

   ```
   install  -n /usr/bin fixit
   ```

   This replaces the **fixit** file if it is found in one of the default directories. If the file is not found, it is installed as **/usr/bin/fixit**.
5. To install a new command, enter:

   ```
   install  -c /usr/bin fixit
   ```

   This creates a new command by installing a copy of the **fixit** file as **/usr/bin/fixit**, but only if this file does not already exist.
6. To install a command in a specified directory whether or not it already exists, enter:

   ```
   install  -f /usr/bin  -o  -s fixit
   ```

   This forces the **fixit** file to be installed as **/usr/bin/fixit** whether or not it already exists. The old version, if any, is preserved by moving it to **/usr/bin/OLDfixit** (a result of the **-o** flag). The messages that tell where the new command is installed are suppressed (a result of the **-s** flag).

Compatibility

For compatibility with Berkeley Software Distribution (BSD), two **install** commands exist. See the **installbsd** command.

Files

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

Related Information

The **chgrp** command, **chmod** command, **chown** command, **cp** command, **installbsd** command, **make** command, **mv** command, **strip** command.

---

install_all_updates Command

Purpose

Updates installed software to the latest level on media and verifies the current recommended maintenance level.

Syntax

install_all_updates -d *Device* [**-p**] [**-i**] [**-c**] [**-r**] [**-n**] [**-s**] [**-x**] [**-v**] [**-N**] [**-S**] [**-Y**] [**-V**] [**-D**]

Description

install_all_updates examines currently installed software and attempts to update it to the latest level that is available on the media. **install_all_updates** will not install any filesets that are present on the media, but not installed on the system, unless these filesets are installed as requisites of other selected filesets. For installp images, all installp requisites are enforced.

Notes:

1. Currently, **install_all_updates** processes installp images and rpm images. Because the **rpm** utility does not support automatic installation of requisites, some rpm software may not be installable with **install_all_updates**.
2. **install_all_updates** verifies the current recommended maintenance level by using the ″**oslevel**″ utility and checking with the latest recommended maintenance level known to this version of **install_all_updates**.
3. If **install_all_updates** locates an update to the install utilities (the **bos.rte.install** fileset), it first installs the update and then reinvokes itself to process the remaining updates. The ″**-i**″ flag can be used to update the install utilities only, this is useful when attempting to view an accurate preview.
4. **install_all_updates applies** all installp updates unless the **COMMIT** flag (**-c**) is specified. For more information of **APPLY** vs. **COMMIT** please see the **installp** man page.
5. **install_all_updates** will by default instruct **installp** to automatically install requisites and to do any necessary filesystem expansions. The ″**-n**″ will override the install requisite default, and ″**-x**″ will override the filesystem expansion default.
6. The following flags apply to installp updates *only*: **-c**, **-n**, **-x**, **-v**, **-S**, and **-V**.

Flags

-c	Instructs **installp** to commit all newly installed updates. Updates are applied by default (Please see the **installp** man page for more explanation on applying vs. committing updates).
-d *Device*	Specifies where the installation media can be found. This can be a hardware device such as tape or cdrom or it can be a directory that contains installation images. When installation media is a tape device it should be specified as no-rewind-on-close and no-retension-on-open.
-D	Turns on **install_all_updates** debug output. This flag is for debugging the **install_all_updates** utility and should not be used for normal operations.
-i	Update install utilities only.

-n	Instructs **installp** to not automatically install requisites. Automatic installation of requisites is the default behavior.
-N	Skip updating install utilities first. **Note:** This flag is not recommended unless you are debugging a related problem.
-p	Performs a preview of an action by running all preinstallation checks for the specified action. No software changes are made.
-r	Update rpm images (if possible). This flag is not set by default.
-s	Skip recommended maintenance level verification. The verification is performed by default.
-S	Instructs **installp** to suppress multi-volume processing of cdrom media.
-v	Instructs **installp** to verify that all installed files in the fileset have the correct checksum value after the installation. This operation may require more time to complete the installation.
-V	Instructs **installp** to run in verbose output mode.
-x	Instructs **installp** to not automatically expand file systems. Automatic expansion of file systems is the default.
-Y	Agrees to all software license agreements which are required for software installation.

Exit Status

0 All **lppmgr** related operations completed successfully.

>0 An error occurred.

Security

Only the root user can execute **install_all_updates**.

Examples

1. To install all installp updates on device **/dev/cd0** and to verify the current recommended maintenance level, enter:
   ```
   install_all_updates -d /dev/cd0
   ```
2. To commit install all installp updates and install any installable rpm updates in directory **/images**, enter:
   ```
   install_all_updates -d /images -rc
   ```
3. To install the latest level of install utilities on device **/dev/cd0** (**bos.rte.install** installp fileset), enter:
   ```
   install_all_updates -d /dev/cd0 -i
   ```

Files

/usr/sbin/install_all_updates	Contains the **install_all_updates** command.

Related Information

The **installp**l command, **lslpp** command, **rpm** commnand

install_assist Command

Purpose

Starts the Installation Assistant application.

Syntax

install_assist

Description

The **install_assist** command starts Installation Assistant, an application designed to simplify the customization of your system after a Base Operating System (BOS) installation. The Installation Assistant guides you through post-installation tasks and, in some cases, automatically installs software packages for you. The Installation Assistant has two interfaces, ASCII and graphical. The interface that displays is based on your terminal type (defined in the **TERM** environment variable).

If your terminal type is not set, the first menu displayed by the ASCII Installation Assistant requires you to enter your terminal type (tty). If you enter a terminal type that is not valid, this menu redisplays until a valid type is entered. If you enter a valid terminal type that does not match your terminal, the next screen displayed could be unreadable. In this case, press the break key sequence to return to the Set Terminal Type screen. For most terminal types, the break key sequence is Ctrl-C.

On a system with an ASCII interface, the newly installed BOS reboots and starts the Installation Assistant to guide you through completing configuration tasks. You must have root user authority to use the Installation Assistant. To access the Installation Assistant later, type **install_assist** on the command line. You can also access it from a graphics system through the SMIT **smit assist** fast path. If there are outstanding software license agreements that must be accepted before you can continue to use the machine, the Installation Assistant prompts you to view and accept these agreements.

On a system with a graphical interface, the newly installed BOS reboots and the Configuration Assistant starts to guide you through the configuration tasks. If there are outstanding software license agreements that must be accepted before you can continue to use the machine, the Configuration Assistant prompts you to view and accept these agreements. To access the Configuration Assistant later, type **configassist** on the command line.

Most Installation Assistant tasks create or add to the **smit.log** and **smit.script** files in your home directory. (These are the same files appended when you run a SMIT session.) The commands built and run by the Installation Assistant tasks are added to the end of the **smit.log** file along with the command output. The time, name of the task, and the command (flags and parameters included) are added to the end of the **smit.script** file in a format that can easily be used to create executable shell scripts.

Example

1. To start the Installation Assistant, type:
   ```
   install_assist
   ```
2. To access the Configuration Assistant, type:
   ```
   configassist
   ```
3. Access the Installation Assistant from a graphical interface, use the SMIT `smit assist` fast path.

Files

smit.log	Specifies detailed information on your session, with time stamps.
smit.script	Specifies the task commands run during your session, with time stamps.

Related Information

The **configassist** command.

Configuring the Operating System in *AIX 5L Version 5.2 Installation Guide and Reference*

---

install_mh Command

Purpose

Sets up mailbox directories.

Syntax

install_mh [**-auto**] [**-help**]

Description

The **install_mh** command sets up mailbox directories. The **install_mh** command is not started by the user. The **install_mh** command is called by other programs only.

The **install_mh** command starts automatically the first time you run any Message Handler (MH) command. The **install_mh** command prompts you for the name of your mail directory. If the directory does not exist, the **install_mh** command queries you if it should be created. Upon receiving a positive response, the **install_mh** command creates the **$HOME/.mh_profile** file and places the `Path:` profile entry in it. This entry identifies the location of your mailbox by specifying the directory path for your MH directory, *UserMHDirectory*.

Flags

-auto Creates the standard MH path without prompting.
-help Lists the command syntax, available switches (toggles), and version information.

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

Files

$HOME/.mh_profile Contains the MH user profile.

Related Information

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

install_wizard Command

Purpose

Invokes the Web-based System Manager Install Wizard or the SMIT install menu.

Syntax

install_wizard [**-d** *Media*]

Description

The **install_wizard** command invokes the Web-based System Manager Install Wizard or the SMIT install menu. This provides an easy path to the install interfaces. These interfaces show media content for **installp**, UDI, or ISJE products and launch the appropriate installer.

Flags

-d *device or directory* The device or directory containing the images to install.

Example

To invoke the Web-based System Manager Install Wizard, insert an install CD in cd1 and type:

```
install_wizard -d /dev/cd1
```

Files

/usr/sbin/install_wizard

Related Information

The **installp** command.

---

installbsd Command

Purpose

Installs a command (BSD version of the **install** command).

Syntax

/usr/bin/installbsd[**- c**][**- g** *Group*] [**- m** *Mode*][**- o** *Owner*][**- s**] *BinaryFileDestination*

Description

The **installbsd** command installs the file specified by the *BinaryFile* parameter by moving it to a file or directory specified by the *Destination* parameter. Use of the **-c** flag copies the *BinaryFile* rather than moving it. If the specified *Destination* parameter is a directory, the *BinaryFile* is moved into the directory. If the specified *Destination* parameter already exists as a file, the **installbsd** command removes that file before the *BinaryFile* is moved. The **installbsd** command does not move a file onto itself.

Installing the file /**dev**/**null** creates an empty file.

Flags

-c Copies the file specified by the *BinaryFile* parameter to the file or directory specified by the *Destination* parameter.

-g *Group* Specifies a group for the file specified by the *Destination* parameter. The default group is staff.

-m *Mode* Specifies a mode for the file specified by the *Destination* parameter. The default mode is 755. The specified mode can be an octal number or a symbolic value.

-o *Owner* Specifies the owner for the file specified by the *Destination* parameter. The default owner is the root user.

-s Causes the file specified by the *BinaryFile* parameter to be stripped after installation.

Examples

To install a new command called **fixit**, enter:

```
installbsd  -c o mike fixit /usr/bin
```

This command sequence installs a new command by copying the program `fixit` to `/usr/bin/fixit`, with user `mike` as the owner.

Files

/usr/ucb/install	Hard-link to the **/usr/bin/installbsd** file.
/usr/bin/installbsd	Contains the **installbsd** command.

Related Information

The **chgrp** command, **chmod** command, **chown** command, **cp** command, **install** command, **mv** command, **strip** command.

---

installp Command

Purpose

Installs available software products in a compatible installation package.

Syntax

To Install with Apply Only or with Apply and Commit

installp [**-a** | **-a c** [**-N**]] [**-e***LogFile*] [**-V** *Number*] [**-d***Device*] [**-E**] [**-Y**] [**-b**] [**-S**] [**-B**] [**-D**] [**-l**] [**-p**] [**-Q**] [**-q**] [**-v**] [**-X**] [**-F** | **-g**] [**-O** { [**r**] [**s**] [**u**] }] [**-t***SaveDirectory*] [**-w**] [**-z***BlockSize*] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To Commit Applied Updates

installp -c [**-e***LogFile*] [**-V***Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To Reject Applied Updates

installp -r [**-e***LogFile*] [**-V***Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* }

To Deinstall (Remove) Installed Software

installp -u [**-e***LogFile*] [**-V***Number*] [**-b**] [**-g**] [**-p**] [**-v**] [**-X**] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f** *ListFile* }

To Clean Up a Failed Installation:

installp -C [**-b**] [**-e***LogFile*]

To List All Installable Software on Media

installp { **-l** | **-L** } [**-e***LogFile*] [**-d** *Device*] [**-B**] [**-l**] [**-q**] [**-E**] [**-z***BlockSize*] [**-O** { [**s**] [**u**] }]

To List All Customer-Reported Problems Fixed with Software or Display All Supplemental Information

installp { **-A**| **-i** } [**-e***LogFile*] [**-d***Device*] [**-B**] [**-l**] [**-q**] [**-z** *BlockSize*] [**-O** { [**s**] [**u**] }] { *FilesetName* [*Level*]... | **-f** *ListFile* | **all** }

To List Installed Updates That Are Applied But Not Committed

installp -s [**-e***LogFile*] [**-O** { [**r**] [**s**] [**u**] }] [**-w**] { *FilesetName* [*Level*]... | **-f***ListFile* | **all** }

To List Platform Specific Installable Software on Media

installp { **-l** | **-L** } { **-M***Platform* } [**-e***LogFile*] [**-d** *Device*] [**-B**] [**-l**] [**-q**] [**-z** *BlockSize*] [**-O** { [**s**] [**u**] }]

Description

Note: The **noclobber** option of the Korn or C shell should be unset in the environment from which an installation is performed.

The **installp** command installs and updates software.

A fileset is the lowest installable base unit. For example, **bos.net.tcp.client 4.1.0.0** is a fileset. A fileset update is an update with a different fix ID or maintenance level. For example, **bos.net.tcp.client 4.1.0.2** and **bos.net.tcp.client 4.1.1.0** are both fileset updates for **bos.net.tcp.client 4.1.0.0**.

When a base level (fileset) is installed on the system, it is automatically committed. You can remove a fileset regardless of the state (committed, broken, committed with applied updates, committed with committed updates, etc.).

When a fileset update is applied to the system, the update is installed. The current version of that software, at the time of the installation, is saved in a special save directory on the disk so that later you can return to that version if desired. After a new version of a software product has been applied to the system, that version becomes the currently active version of the software.

Updates that have been applied to the system can be either committed or *rejected* at a later time. The **installp -s** command can be used to get a list of applied updates that can be committed or rejected.

When updates are committed with the **-c** flag, the user is making a commitment to that version of the software product, and the saved files from all previous versions of the software product are removed from the system, thereby making it impossible to return to a previous version of the software product. Software can be committed at the time of installation by using the **-ac** flags. Note that committing already applied updates does not change the currently active version of a software product. It merely removes saved files for previous versions of the software product.

When a base level is removed with the **-u** flag, the files that are part of the software product and all its updates are removed from the system. Most cleanup of system configuration information pertaining to the product is also done, but this is dependent on the product and may not always be complete.

When a software product update is rejected with the **-r** flag, the active version of the software product is changed to the version immediately previous to the rejected update. Files saved for the rejected update and any updates that were applied after it are removed from the system.

A software product that is to be removed from the system can be in any state. Any product updates can be in either the applied or committed state, and they will also be removed.

If a previously interrupted installation leaves any software in a state of either applying or committing, it is necessary to perform cleanup with the **-C** flag before any further installations will be allowed. Although the **installp -C** command accepts software product names on the command line without returning an error, an attempt is always made to clean up all products when the **-C** flag is used. An attempt is made to clean up any incomplete installations by removing those parts that were previously completed. An attempt is also made to return to the previous version of the software product, if one exists, as the currently active version. If this cannot be done, the software product is marked as *broken*, and unpredictable results can occur if the user attempts to use it. Therefore, it is advisable for the user to reinstall any broken software products or updates.

The **-t** flag specifies an alternate location for a save directory that holds files being replaced by an update. This option is primarily useful in the following two circumstances.

- You have enough local disk space for saving replaced files but you do not want to permanently expand the root and **/usr** file systems.

  In this case, you can choose to create a separate file system for the alternate save directory. When you are satisfied with the updated system and have committed all applied updates, disk space can be retrieved by deleting the save file system.
- If you do not have enough local disk space for saving replaced files but you have access to ample disk space on a remote system, then you can specify a directory that is mounted from a remote file system.

  If a remote file system is used, commit the updates *as soon as possible*. You may want to initiate the installation action as an `apply` and `commit` operation with the **-ac** flags. If you want to `apply` only to be capable of rejecting any unwanted updates, then test the newly installed updates *as soon as possible* and then commit or reject them.

  Take into account the following considerations when using an alternate save directory:
- It is recommended that you use the same alternate save location on each invocation of the **installp** command.
- If an alternate save directory is used for an apply operation, make sure that the file system containing that directory remains mounted. It is highly recommended that any necessary mounts be done automatically on a reboot.
- If an alternate save directory is missing on a commit operation, the commit takes place, and a warning is given stating that the save directory could not be deleted. It is then your responsibility to delete the save directories that are no longer used in order to retrieve that disk space.
- If an alternate save directory is missing on reject, the reject operation cannot be done because the saved files are missing. An error is given, and the entire reject operation is cancelled. If the missing save directory is not caused by a temporary situation (for example, the inability to contact a remote directory on the network,) your only options are to commit the updates or leave them in an applied state permanently.
- When doing a system backup, you are responsible for backing up any alternate save directories that do not reside in the root volume group.
- The installation process safeguards users with a remote save directory from the possibility of two different systems using the same remote directory. However, use directory pathnames that easily and uniquely identify each user's system. For example, you might add the system's hostname somewhere in the pathname.
- Do not create a **mksysb** backup of a system with a remote save directory and then try to restore the **mksysb** image onto a system other than the original. In this case, using a **mksysb** image to install several like systems causes multiple ownership of the same remote save directory.

The **installp -A** command can be used to obtain a list of the Authorized Program Analysis Report (APAR) numbers and summaries for all customer-reported problems that are fixed in the specified software package. The **installp -i** command can be used to display supplemental information contained in files that can be a part of the specified software package.

To list all the software products and updates on the specified installation media, use the **installp -l** command. The output of the **installp** command with the **-l** flag resembles the following:

```
# Fileset Name            Level                  I/U Q Content
#==============================================================
  X11.adt.include         4.1.0.0                I  N usr
#   AIXwindows Application Development Toolkit Include F

  X11.adt.lib             4.1.0.0                I  N usr
#   AIXwindows Application Development Toolkit Libraries
#
  X11.adt.motif           4.1.0.0                I  N usr
#   AIXwindows Application Development Toolkit Motif
```

```
#
  X11.adt.bitmaps          4.1.0.0                  I  N usr
#   AIXwindows Application Development Toolkit Bitmap Fi

#
  X11.adt.ext              4.1.0.0                  I  N usr
#   AIXwindows Application Development Toolkit for X Ext
#
  X11.adt.imake            4.1.0.0                  I  N usr
#   AIXwindows Application Development Toolkit imake
#
  X11.apps.rte             4.1.0.0                  I  N usr
#   AIXwindows Runtime Configuration Applications
#
  X11.apps.msmit           4.1.0.0                  I  N usr
#   AIXwindows msmit Application
```

The fields have the following meanings:

`Fileset Name` Name of the fileset to be installed.

`Level` Level of the fileset to be installed.

`I/U` Indicates the type of package of which the fileset is a part. The fileset may belong to an installation package or to one of several types of update packages. The package types are as follows:

- **I** Indicates an installation package.
- **S** Indicates a single update.
- **SR** Indicates a required update. Whenever the **installp** command encounters a required update, the update is automatically included in the input list.
- **SF** Indicates a required update. Whenever the **installp** command encounters a required update, the update is automatically included in the input list. Reserved for updates to the **installp** fileset.
- **M** Indicates a maintenance package. This is a packaging update that contains only a list of other updates to be applied. This package delivers no files.
- **ML** Indicates an update package that identifies a new maintenance level for the product. This is a cumulative set of all updates since the previous product level.

`Q` Quiescent (quiet) column. A **Y** indicates that running processes can be affected by the installation of this fileset. Refer to the documentation supplied with the software product. An **N** indicates that running processes are not affected by the installation of this fileset. A **B** indicates **bosboot** and quiescent. A **b** indicates **bosboot** and not quiescent.

`Content` Content column:

- **usr,root** **/usr** and root file systems (AIX 3.2 and later)
- **usr** **/usr** file system only (AIX 3.2 and later)
- **share** **/usr/share** file system only (AIX 3.2 and later)

Output from the **installp -s** command, which is used to get a list of applied software fileset updates and updates that are available to be either committed or rejected, resembles the following:

```
Installp Status
---------------
Name                      Part    Level                    State
-------------------------------------------------------------------
bos.net.tcp.client        USR     4.1.0.2                  APPLIED
bos.net.tcp.client        ROOT    4.1.0.2                  APPLIED
```

```
bos.rte.commands          USR     4.1.0.1                 APPLIED
bos.rte.misc_cmds         USR     4.1.0.1                 APPLIED
bos.rte.tty               USR     4.1.0.1                 APPLIED
```

The fields have the following meanings:

Name	Name of the installed software product fileset.
Part	The part of the fileset where:
ROOT	root file system
SHARE	**/usr/share** file system
USR	**/usr** file system.
Level	The level of the installed software product option.
State	The state of the installed software product option.

The software products and updates to be installed can be identified in one of three ways:

- by the keyword **all**, which indicates that all software contained on the specified installation media is to be installed
- by a list of software product names (each of which can optionally be followed by a level) that indicates the software to be installed
- by the **-f** flag followed by a file name, where each line in the file is an entry containing a software product name, optionally followed by a level, or is a comment line that begins with a # and is ignored

Note: The **installp** program uses the **sysck** command to verify files after restoring them. The **sysck** command does not recognize the following special characters in file names: ~, `, ', \, ″, $, ^, &, ( ), |, {}, [], <>, and ?. If a file name contains one of these characters, installation fails.

The *FilesetName* parameter can be used to specify an entire software product or any separately installable filesets within the software package. For example, **bos.net** is the name of a software package, and the separately installable filesets within that software package are **bos.net.ncs.client**, **bos.net.nfs.client**, and **bos.net.tcp.client**. If the user specifies **bos.net** for the *FilesetName* parameter, then all of the separately installable filesets listed are installed. If the user specifies **bos.net.tcp.client** for the *FilesetName* parameter, then only that fileset is installed.

The *Level* parameter indicates the level of the software product or update that is to be installed. The *Level* parameter is of the form *vv.rr.mmmm.ffff.ppppppppp* where:

vv	is a numeric field of 1 to 2 digits that represents the version number.
rr	is a numeric field of 1 to 2 digits that represents the release number.
mmmm	is a numeric field of 1 to 4 digits that represents the modification level.
ffff	is a numeric field of 1 to 4 digits that represents the fix level.
ppppppppp	is a character field of 1 to 9 characters that represents the fix ID.

If a user is installing an installation package from installation media that contains only installation packages it is not usually necessary to specify the level. More than one software product installation package with different levels does not often exist on the same installation medium, but when this does occur **installp** installs the specified software product at the latest software product level when *Level* is not specified with *FilesetName*. For installation media that contain either update packages only or contain both installation and update packages, all applicable update packages that are present on the installation media for the specified *FilesetName* are also installed when *Level* is not specified. For installation media that contain both installation and update packages the user can request the installation of only installation packages or only update packages by specifying the **-I** or **-B** flags, respectively. If the user wants to install only some of the updates on the installation medium for a specific software product both *FilesetName* and *Level* for each of the updates to be installed for that software product must be specified.

An example of what might be entered to install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images** directory would be the following:

```
installp -a -d/usr/sys/inst.images bos.net.tcp.client 4.1.0.0
bos.net.tcp.client 4.1.0.2
```

Note: In the event that there are duplicate filesets at the same level, **installp** will use the first one that it finds in the install table of contents (**.toc**). This situation can occur when **bffcreate** is used to extract images from different media to the same installation directory. For this reason, make sure that update images are not extracted to the same directory as base level images for the same fileset at the same level.

A summary report is given at the end of the **installp** output that lists the status of each of the software products that were to be installed. An example summary report for the previous **installp** command follows:

```
Installp Summary
----------------
Name                    Level        Part        Event        Result
-------------------------------------------------------------------
bos.net.tcp.client      4.1.0.0      USR        APPLY        SUCCESS
bos.net.tcp.client      4.1.0.0      ROOT       APPLY        SUCCESS
bos.net.tcp.client      4.1.0.2      USR        APPLY        SUCCESS
```

Note: If a previously installed level of a fileset update is in the broken state, the **-acgN** flags must be used when that fileset update is installed again.

Summary Report Values

The summary report identifies the name of the product option and the part of the product. Other information given includes the requested action (event) and the result of that action.

Event Values

The Event column of the summary report identifies the action that has been requested of the **installp** command. The following values may be found in this column:

Event	Definition
APPLY	An attempt was made to apply the specified fileset.
COMMIT	An attempt was made to commit the specified fileset update.
REJECT	An attempt was made to reject the specified fileset update.
CLEANUP	An attempt was made to perform cleanup for the specified fileset.
DEINSTALL	An attempt was made to remove the specified fileset.

Result Values

The Result column of the summary report gives the result of **installp** performing the requested action. It can have the following values:

Result	Definition
SUCCESS	The specified action succeeded.
FAILED	The specified action failed.
CANCELLED	Although preinstallation checking passed for the specified option, it was necessary to cancel the specified action before it was begun. Interrupting the installation process with Ctrl-c can sometimes cause a canceled action, although, in general, a Ctrl-c interrupt causes unpredictable results.

Flags

-A Displays the APAR number and summary of all customer-reported problems that are fixed in the specified software package. No installation is attempted.

-a Applies one or more software products or updates. This is the default action. This flag can be used with the **-c** flag to apply and commit a software product update when installed.

-b Prevents the system from performing a bosboot in the event that one is needed.

-B Indicates that the requested action should be limited to software updates.

-C Cleans up after an interrupted installation and attempts to remove all incomplete pieces of the previous installation. Cleanup should be performed whenever any software product or update is in a state of either *applying* or *committing* and can be run manually as needed. For backwards compatibility other flags and parameters can be accepted with **installp -C**, but are ignored because all necessary cleanup is attempted.

-c Commits all specified updates that are currently applied but not committed. When an update is committed all other software products it is dependent on must also be committed (unless they are already in the committed state). The specified software product is dependent on any software product that is a prerequisite or corequisite of the specified product. The commit will fail and error messages will be given if any requisite software products are not in the committed state. The **-g** flag can be used to automatically commit requisite software product updates.

-D Deletes the installation image file after the software product or update has been successfully installed. When the **-g** flag is specified, the installation image files for any products that are automatically included will also be deleted. This flag is valid only with the **-a** or **-ac** flags and is not valid with the **-Or** flag. This flag is also only valid when the device is a directory and an installation image file on the system where the installation is taking place.

-d *Device* Specifies where the installation media can be found. This can be a hardware device such as tape or diskette, it can be a directory that contains installation images, or it can be the installation image file itself. When the installation media is a product tape or Corrective Service tape, specified the tape device as no-rewind-on-close and no-retension-on-open. Examples of this would be **/dev/rmt0.1** for a high density tape, or **/dev/rmt0.5** for a low density tape. Use the options specified by the tape supplier. The default device is **/dev/rfd0**.

-e *LogFile* Enables event logging. The **-e** flag enables the user to append certain parts of the **installp** command output to the file specified by the *LogFile* variable. By default the output of the **installp** command goes to **stdout** and **stderr**, unless SMIT or VSM is used, in which case the output goes to the **smit.log**. The *LogFile* variable must specify an existing, writable file, and the file system in which the file resides must have enough space to store the log. The log file does not wrap.

Not all output is appended. Copyright information is still displayed to the user. Any error messages are displayed on the screen and sent to the file specified by the *LogFile* variable. A results summary of the **installp** command invocation is also displayed on the screen and sent to the *LogFile*. This flag is primarily used by NIM and BOS install to limit the output shown to the user, but keep useful information for later retrieval.

-E Displays software license agreements. This flag is only valid with the **-a** or **-l** flags. If the **-E** flag is specified with the **-a** flag, a new section is emitted showing the pending license agreements associated with the selected filesets. If the **-E** flag is specified with the **-l** flag, output is emitted showing the license agreements associated with all filesets on the media.

-F This option can be used to force the installation of a software product even if there exists a previously installed version of the software product that is the same as or newer than the version currently being installed. The **-F** flag is not valid with update packages or the **-g** flag. When you use the **-F** flag, the **-l** flag is implicit.

-f *ListFile* Reads the names of the software products from *ListFile*. If *ListFile* is a - (dash), it reads the list of names from the standard input. Software fileset names, optionally followed by a level, should be one per line of text, and any text following the second set of white spaces or tabs on a line is ignored. Output from the **installp -l** command is suitable for input to this flag.

-g When used to install or commit, this flag automatically installs or commits, respectively, any software products or updates that are requisites of the specified software product. When used to remove or reject software, this flag automatically removes or rejects dependents of the specified software. The **-g** flag is not valid when used with the **-F** flag.
Note: This flag also automatically pulls in a superseding update present on the media if the specified update is not present. This flag causes the newest update to be installed for a given fileset, when there are multiple superseding updates for the same fileset on the installation media.

-I (uppercase i) Indicates that the requested action should be limited to base level filesets.

-i Displays on standard output the **lpp.instr**, **lpp.doc**, **lpp.README**, and **README** files on the installation media for the software product, if they exist. This flag can take a significant amount of time for a large number of filesets.

-J This flag is used when the **installp** command is executed from the System Management Interface Tool (SMIT) menus.

-l (lowercase L) Lists all the software products and their separately installable options contained on the installation media to standard output. No installation occurs. The **-l** flag is not valid with the **-Or** flag.

-L Displays the contents of the media by looking at the table of contents (TOC) and displaying the information in colon-separated output. This flag is used by smit and vsm to list content of the media. The format provided:

```
package:fileset:v.r.m.f:PTF:type:state:supersede:\
sup_ptf:sup_state:latest_sup:quiesce:Descr:\
netls_vendor_id:netls_prod_id:netls_prod_ver
```

-M*Platform* Specifies the *Platform* value. Any of the following values may be used to list the installable software packages:

R Specifies POWER-based platform packages only.

N Specifies neutral packages, that is, packages that are not restricted to the POWER-based platform.

A Specifies all packages.

-N Overrides saving of existing files that are replaced when installing or updating. This flag is valid only with the **-ac** flags. If there is a failure in the system during the installation, there is no recovery of replaced files when this flag is used.

-O{[**r**][**s**][**u**]} Installs the specified part of the software product. The **r** indicates the **/** (root) part is to be installed, the **s** indicates the **/usr/share** part is to be installed, and the **u** indicates the **/usr** part is to be installed. The **-O** flag is not needed with standard systems because without this flag all parts are installed by default. This flag is needed for use with the installation of diskless or dataless workstations and is designed for use by the **nim** command. The **-Or** option is not valid with the **-d** or **-l** flags.

-p Performs a preview of an action by running all preinstallation checks for the specified action. This flag is only valid with apply, commit, reject, and remove (**-a**, **-c**, **-r**, and **-u**) flags.

-Q Suppresses errors and warnings concerning products failing to install due to instrequisites. This flag applies only to AIX 4.2 or later.

-q Specifies quiet mode, which suppresses the prompt for the device, except for media volume change.

-r Rejects all specified software updates that are currently applied but not committed. When a software update is rejected any other software product that is dependent on it (that is, those software products that have the specified software product as a requisite) must also be rejected. The **-g** flag can be used to reject automatically dependent software updates. The keyword **all** is not valid with the reject flag (**-r**). For backwards compatibility, the **-R** flag is also accepted as a reject flag. The **-R** cannot be used to remove base level filesets; use the **-u** flag.

-s Lists information about all software products and updates that have been applied but not committed. This list comprises the software that is available to be either committed or rejected.

-S Suppresses multiple volume processing when the installation device is a CD-ROM. Installation from a CD_ROM is always treated as a single volume, even if the CD-ROM contains information for a multiple volume CD set. This same suppression of multiple volume processing is performed if the **INU_SINGLE_CD** environment is set.

-t *SaveDirectory* Specifies an alternate save directory location for files being replaced by an update.

The **-t** flag is only valid with an apply or an apply/commit operation for updates. This flag is not valid with the **-N** flag.

The **-t** flag is useful when there is insufficient space in the default file systems (**/** and **/usr**) or when it is undesirable to permanently expand these file systems. It may be desirable for the specified directory to be a remote file system. A remote file system must have ample space, because the **installp** command cannot expand remote file systems.

-u Removes the specified software product and any of its installed updates from the system. The product can be in either the committed or broken state. Any software products that are dependent on the specified product must also be explicitly included in the input list unless the **-g** flag is also specified. Removal of any **bos.rte** fileset is never permitted.

-v Verifies that all installed files in the fileset have the correct checksum value after the installation. Installed files are always verified for correct file size after installation. Use this flag after network or remote device installations. If any errors are reported, it might be necessary to install the software product again. Post-installation requisite consistency checks are also started by this flag.

-V *Number* Specifies the verbose option that provides four levels of detail for preinstallation output. The valid values for the *Number* parameter are 2, 3, or 4. The default level of verbosity, without the use of the -V flag, prints an alphabetically ordered list of FAILURES, WARNINGS, and SUCCESSES from preinstallation processing. Requisite failures are reported with emphasis on the real cause of the failure. Extraneous requisites for failed filesets are not displayed. The preinstallation output is modified by levels 2 through 4 as described below:

2 Prints alphabetically ordered list of FAILURES and WARNINGS. Requisite failures are displayed with additional information describing requisite relationships between selected filesets and the requisites causing them to fail. Failing requisites suppressed under *Level 1*are displayed. Preinstallation SUCCESSES are displayed in the order in which they are processed.

3 Level 3 is the same as Level 2, with the exception that additional requisite information is displayed for SUCCESSES.

4 Level 4 is the same as Level 3 for SUCCESSES and WARNINGS. Requisite failures are displayed in a format depicting detailed requisite relationships.

Note: If verbosity level 2 or higher is used, the files that are restored on to the system is shown in the output. Because this will make **installp**'s output much more verbose, make sure that your **/** (root) filesystem does not become full when the /smit.log becomes large (if using smit to run installp).

-w Does not wildcard *FilesetName*. Use this flag from smit so it only installs the fileset chosen and will not install filesets that match. For example, if you choose `foo.rte`, `foo.rte.bar` is not automatically pulled in, as it would be by default, without the **-w** flag. This flag applies only to AIX 4.2 or later.

-X Attempts to expand any file systems where there is insufficient space to do the installation. This option expands file systems based on current available space and size estimates that are provided by the software product package. Note that it is possible to exhaust available disk space during an installation even if the **-X** flag is specified, especially if other files are being created or expanded in the same file systems during an installation. Also note that any remote file systems cannot be expanded.

-Y	Agrees to required software license agreements for software to be installed. This flag is only valid with the **-a** flag.
-z *BlockSize*	Indicates in bytes the block size of the installation media. The default value of Size is 512.
FilesetName	This is the name of the software product to be installed and can specify either an entire software product or any separately installable filesets within the software product. This can be used to specify the name of a fileset or fileset update.
Level	This indicates the level of the software product or update that is to be installed and is of the form vv.rr.mmmm.ffff. If a fileset update has an additional fix ID (also know as ptf id), that ID should also be specified in the Level as in vv.rr.mmmm.ffff.ppppppp.

Return Values

A zero (0) return value indicates that all attempted installations were successful, or that no processing was required for the requested action on the requested filesets (for example, if a requested fileset was already installed).

A nonzero return value indicates that some part of the installation was not successful.

A summary report is given at the end of the **installp** output that lists the status of each of the software products that were to be installed. For those software products that could not be installed or whose installation failed, the user can search for the cause in the more detailed information that is continually displayed from the **installp** command during the installation process.

Security

Privilege Control: Only the root user can run this command.

Auditing Events:

Event	Information
INSTALLP_Inst	Success or failure of the apply, commit, reject, and cleanup operations.

Examples

1. To list all software products and installable options contained on an installation cartridge tape, type:
   ```
   installp -L -d /dev/rmt0.1
   ```
2. To list all customer-reported problems fixed by all software products on an installation tape, type:
   ```
   installp -A -d /dev/rmt0.1 all
   ```
3. To install (automatically committed) all filesets within the **bos.net** software package (located in the **/usr/sys/inst.images** directory) and expand file systems if necessary, type:
   ```
   installp -aX -d/usr/sys/inst.images bos.net
   ```
4. To reinstall and commit the NFS software product option that is already installed on the system at the same level (from tape), type:
   ```
   installp -acF -d/dev/rmt0.1 bos.net.nfs.client 4.1.0.0
   ```
5. To install (apply only) certain updates that are contained on diskette for the TCP/IP software product, type:
   ```
   installp -a bos.net.tcp.client 4.1.0.2 bos.net.tcp.server 4.1.0.1
   ```
6. To remove a fileset named `bos.net.tcp.server`, type:
   ```
   installp -u bos.net.tcp.server
   ```
7. To specify an alternate storage directory on a remote file system for a BOSNET TCP/IP update with **-t/temp_space**, see the following example: the save directory becomes **/temp_space/***My_Hostname***/usr/lpp/bos.net/bos.net.nfs.client/4.1.1.0.save**.

```
mount Server_Name:/Save_Area /temp_space

installp -a -t /temp_space/My_Hostname  \
bosnet.nfs.client 4.1.1.0
```

8. In order to capture a log file of all output from the **installp** command, the **script** command can be used as in the following example. Output is written to the **typescript** file in the current directory.

```
script
installp ...
<Ctrl>d
```

or

```
installp ... 2>&1 | tee /tmp/inst.out
```

In the second example, output is written to the screen and a copy is saved.

9. To preview (without performing) the installation of the Application Developer bundle of software using the **installp** command, type:

```
installp -pacgXd /dev/rmt0.1 -f /usr/sys/inst.data/sys_bundles \
/App_Dev.bnd
```

10. To install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images**, type:

A summary report is given at the end of the installp output that lists the status of each of the software products that were to be installed. An example summary report for the previous installp command follows:

```
Installp Summary
----------------
Name                 Level        Part       Event        Result
-------------------------------------------------------------------
bos.net.tcp.client   4.1.0.0      USR        APPLY        SUCCESS
bos.net.tcp.client   4.1.0.0      ROOT       APPLY        SUCCESS
bos.net.tcp.client   4.1.0.2      USR        APPLY        SUCCESS
```

Note: This summary is also saved in **/var/adm/sw/installp.summary** until the next **installp** invocation. The header file **inuerr.h** in the **/usr/include** directory describes the fields making up the records in the **installp.summary** file.

11. To list software products (located in the **/usr/sys/inst.images** directory) that are installable on POWER-based machines, type:

```
installp -l -MR -d /usr/sys/inst.images
```

Files

/dev/rfd0	Specifies the default restore device.
/dev/rmt*n*	Specifies the raw streaming tape interface.
/usr/sys/inst.images directory	Contains files in backup format for use in installing or updating a complete set or subset of software products.

Related Information

The **bffcreate** command, **inudocm** command, **inutoc** command, **lppchk** command, **lslpp** command, **sysck** command.

installms Command

Purpose

installms - Installs the Cluster Systems Management (CSM) management server software.

Syntax

installms [**–h**]

installms [**–f**] [**–v** | **–V**] {**–x** | **–p** *pkg_path1:pkg_path2*} [*Attr=value* [*Attr=value*...]]

Description

The **installms** command installs the Cluster Systems Management (CSM) management server software. It automatically installs both CSM and its prerequisite software. Before running the **installms** command, a supported version of the operating system must be installed.

The command performs the following tasks:

- Creates the **/csminstall** directory structure that is required for CSM files.

  Note: If the **/csminstall** directory does not exist, it will be created by the **installms** command. However, it is recommended that you create the **/csminstall** directory in a separate file system before running the **installms** command. See the *IBM CSM for AIX 5L: Software Planning and Installation Guide* for more information on requirements for **/csminstall**.
- Optionally copies the required CSM file sets to the correct **/csminstall** subdirectories.
- Copies the CSM scripts and files that are required to install CSM on the nodes of the cluster to the correct **/csminstall** subdirectories.
- Sets global cluster system attribute values.

If the **–p** flag is used, the **installms** command copies the required file sets from the specified location into the correct subdirectories of **/csminstall**. If the **–x** flag is used, the **installms** command assumes that the files have already been copied. When the **installms** command installs the management server software, it uses the file sets that have been copied to the **/csminstall** subdirectories. For a list of required file sets, see the *IBM CSM for AIX 5L: Software Planning and Installation Guide*.

Flags

–f Forces installation regardless of the status of the existing installation.

–h Writes the command's usage statement to standard output.

–p *pkg_path*
Specifies one or more colon-separated directories where installation packages are located. This flag cannot be used with **–x**.

–v | –V
Writes the command's verbose messages to standard output.

–x Specifies to not copy software packages and to use packages in the **/csminstall** subdirectory. This flag cannot be used with **–p**.

Parameters

*Attr=value***...**
The following attributtes change the *Attr=value* pairs that are stored by the **csmconfig** command.

*RemoteShell***=***full_path_name_of_shell*
The *RemoteShell* attribute value determines the default remote shell for the **dsh** command if the DSH_REMOTE_CMD variable is not set. It also determines which shell to set up on the nodes during installation. The default attribute value for the management server is **/usr/bin/rsh**.

*SetupRemoteShell***={yes | no}**
The *SetupRemoteShell* attribute value controls whether the remote shell is set up on the

management server and the nodes. If it is set to **yes** or **1**, then the remote shell is set up. The remote shell is not set up if the attribute is set to **no** or **0**. The default value for this attribute is **1**.

Security

The command requires root access to the cluster management server.

Exit Status

0 The command has run successfully.

1 An error has occurred.

Environment Variables

DSH_PATH
Sets the path that is used on the remote nodes. If DSH_PATH is not set, the default path for the remote shell is used. For example, **DSH_PATH=$PATH** sets the path on the remote node to the same path that is used on the source node.

DSH_REMOTE_CMD
Specifies the path of the remote shell executable to use instead of the default.

DSH_REMOTE_OPTS
Includes the options specified in the remote command when the command is forwarded to the remote nodes.

DSH_FANOUT
Sets the maximum number of concurrent remote shell commands. This can also be set by the **–f** flag.

DSH_LIST
Specifies a file that contains definitions of the set of nodes that comprise the node list.

DSH_TIMEOUT
Specifies the time, in seconds, to wait for response from the remote shell. This can also be set with the **–t** flag.

WCOLL
Specifies a file that contains definitions of the set of nodes that comprise the working collective.

Standard Error

When the **dshbak** filter is used and standard error messages are generated, all error messages on standard error appear before all standard output messages. This is true with and without the **-c** flag.

Examples

1. To install CSM management server software using CSM file sets that were previously copied to the **/tmp/csm** directory, enter:

   ```
   installms -p /tmp/csm
   ```

2. To install CSM and the packages contained in the current directory on the management server using the force option, enter:

   ```
   installms -f -p .
   ```

3. To install CSM on the management server using the AIX CSM version 1.3.0 packages previously copied to the **/csminstall/AIX/csm/1.3.0/packages** directory, enter:

   ```
   installms -x
   ```

Location

/opt/csm/bin/installms

Files

/var/log/csm/installms.log

Location of the log file on the management server for the **installms** command. Up to five copies of this log are maintained. Old logs receive a numeric suffix up to 4. The oldest file is **installms.log.4**.

Related Information

- The **csmconfig**, **definenode**, **uninstallms**, and **updatenode** commands.
- The *IBM CSM for AIX 5L: Software Planning and Installation Guide*.

---

instfix Command

Purpose

Installs filesets associated with keywords or fixes.

Syntax

instfix [**-T** [**-M** *Platform*]] [**-s** *String*] [**-S**] [**-k** *Keyword* | **-f** *File*] [**-p**] [**-d** *Device*] [**-i** [**-c**] [**-q**] [**-t** *Type*] [**-v**] [**-F**]] [**-a**]

Description

The **instfix** command allows you to install a fix or set of fixes without knowing any information other than the Authorized Program Analysis Report (APAR) number or other unique keywords that identify the fix.

Any fix can have a single fileset or multiple filesets that comprise that fix. Fix information is organized in the Table of Contents (TOC) on the installation media. After a fix is installed, fix information is kept on the system in a fix database.

The **instfix** command can also be used to determine if a fix is installed on your system.

Note: Return codes for the **instfix** command are documented in the **/usr/include/inuerr.h** file, which is shipped with the bos.adt.include fileset. There is also a general failure code of 1 and a single reference to EACCES (13) from **/usr/include/errno.h**.

Flags

-a Displays the symptom text associated with a fix. Can be combined with the **-i**, **-k**, or **-f** flag.

-c Displays colon-separated output for use with **-i** flag. Output includes keyword name, fileset name, required level, installed level, status, and abstract. Status values are:

- **-** Down level
- **=** Correct level
- **+** Superseded
- **!** Not installed

-d *Device* Specifies the input device. Required for all flags except **-i** and **-a**.

-F Returns failure unless all filesets associated with the fix are installed.

-f *File* Specifies the input file containing keywords or fixes. Use **-** (dash) for standard input. The **-T** flag produces a suitable input file format for **-f**.

-i Displays whether fixes or keywords are installed. Use this flag with the either the **-k** or the **-f** flag. Installation is not attempted when the **-i** flag is used. If you do not specify the **-k** or the **-f** flag, all known fixes are displayed.

-k *Keyword* Specifies an APAR number or keyword to be installed. Multiple keywords can be entered. A list of keywords entered with the **-k** flag must be contained in quotation marks and separated with spaces.

-M *Platform* Specifies that any of the *Platform* values may be used to list the fixes for that particular platform.

R Specifies POWER-based platform fixes only.

N Specifies neutral fixes, that is, fixes that are not restricted to the POWER-based platform.

A Specifies all fixes.

-p Displays filesets associated with keywords. This flag is used with either the **-k** or the **-f** flag. Installation is not attempted when the **-p** flag is used.

-q Specifies quiet mode. Use this flag with the **-i** flag. If you use the **-c** flag, no heading is displayed, otherwise there is no output.

-s *String* Searches for and displays fixes on media containing a specified string.

-S Suppresses multiple volume processing when the installation device is a CD-ROM. Installation from a CD_ROM is always treated as a single volume, even if the CD-ROM contains information for a multiple volume CD set. This same suppression of multiple volume processing is performed if the **INU_SINGLE_CD** environment is set.

-T Displays the entire list of fixes present on the media.

-t*Type* Used with the **-i** flag to limit searches to a given type. Valid types are:

f fix

p preventive maintenance

-v Used with the **-i** flag to specify verbose mode. Displays information about each fileset associated with a fix or keyword.

Security

Privilege Control: You must be the root user to install using the **instfix** command, but any user can run the **instfix** command to query the fix database.

Examples

1. To install all filesets associated with fix `IX38794` from the tape mounted on `/dev/rmt0.1`, type:

   ```
   instfix  -k IX38794  -d /dev/rmt0.1
   ```

2. To install all fixes on the media in the tape drive, type:

   ```
   instfix  -T  -d /dev/rmt0.1 | instfix  -d /dev/rmt0.1  -f-
   ```

   The first part of this command lists the fixes on the media, and the second part of this command uses the list as input.

3. To list all keyword entries on the tape containing the string `SCSI`, type:

   ```
   instfix  -s SCSI  -d /dev/rmt0.1
   ```

4. To inform the user on whether fixes `IX38794` and `IX48523` are installed, type:

   ```
   instfix  -i  -k "IX38794 IX48523"
   ```

5. To create a list of filesets associated with fix `IX12345` for `bffs` in the `/bffs` directory, type:

   ```
   instfix  -p  -k IX12345  -d /bffs | installp -acgX -f- -d /bffs
   ```

This sequence passes the list of fixes to the **installp** command to be applied and committed. The **installp** command extends filesystems as needed with the flags shown. This example shows that you can select other **installp** flags. The **instfix** command calls **installp** if the **-p** flag is not used.

6. To list all of the fixes that are not restricted to the POWER-based platform, type:

```
instfix  -T  -MN  -d /dev/cd0
```

Files

/usr/sbin/instfix	Contains the **instfix** command.
/usr/lib/objrepos/fix	Specifies the path to the Object Data Manager database.

Related Information

The **installp** command.

---

inucp Command

Purpose

Performs simple copy operations for the **installp** command. This command is used by the **installp** command and the install scripts.

Syntax

inucp -s *StartDirectory* [**-e** *FinalDirectory*] *ListFile ProductName*

Description

The **inucp** command copies the files in a file tree with its root at *StartDirectory* to the appropriate place on the *FinalDirectory* root.

Before replacing files that may already exist in the *FinalDirectory* file tree, the **inusave** command should be called to save the files until needed by the **inurecv** command.

The *ListFile* parameter specifies a list, one per line, of all the files for *ProductName*. *ListFile* is the full path name of the file that contains the relative path names of files that the product needs to have copied.

The *ProductName* parameter specifies the name of the software product to be copied.

Flags

-e *FinalDirectory*	Indicates the root of the file tree that the files are to be copied to. The *FinalDirectory* should be the base of the file tree. The default directory is the **/** (root) directory when this flag is not specified.
-s *StartDirectory*	Indicates the root of the file tree that the files are to be copied from.

Environment Variables

INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the copy (that is, the **-X** flag was passed). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.
INUTEMPDIR	This flag is set by the **installp** command to the path of the current temporary directory. If this flag is not set the default is **/tmp**.

Error Codes

The **inucp** command returns the following error codes, which are defined in **inuerr.h**.

INUACCS	One or both of *StartDirectory* and *FinalDirectory* are not directories.
INUBADAR	Could not archive files in **lpp.acf** file.
INUBADC1	The copy operation failed.
INUBADMN	Unrecognizable flag specified.
INUGOOD	No error conditions occurred.
INUNOAP2	Could not access the *ListFile*.
INUNODIR	No write access to *FinalDirectory*.
INUNOLPP	One or both of *StartDirectory* and *FinalDirectory* do not have the necessary permissions.
INUNOMK	Could not create a needed directory.
INUNOSPC	Insufficient space for the copy and **INUEXPAND** was not set.
INUTOOFW	One or more parameters were missing.
INUTOOMN	Too many parameters were specified.

Security

Privilege Control: You must be the root user to run this command.

Examples

To copy all the files listed in the **/usr/lpp/X11/inst_root/al** list from the **/usr/lpp/X11/inst_root** file tree to the root directory, enter:

```
inucp  -s /usr/lpp/X11/inst_root /usr/lpp/X11/inst_root/al X11
```

Related Information

The **installp** command, **inurecv** command, **inurest** command, **inusave** command.

---

inudocm Command

Purpose

Displays contents of files containing supplemental information.

Syntax

inudocm [**-d** *Device*] [**-q**] { *ProductName* ... | **all** }

Description

> **Note:** This command is used by the **installp** command and is not recommended as a way to get README information.(See **installp -i**.)

The **inudocm** command is used to display supplemental information. The files from the media that are displayed, if they exist, are the **lpp.doc** file, the **lpp.instr** file, the **lpp.README** file and the **README** file.

The *ProductName* parameter specifies the name of the software product being checked. Specify **all** to display information about all software products that are known to the system.

Flags

-d *Device*	Specifies where the installation media can be found. The *Device* parameter can specify a hardware device, such as a tape or diskette drive, a directory that contains installation images, or an installation image file. The default device is **/dev/rfd0**.

-q Specifies quiet mode, which suppresses prompts.

Security

Privilege Control: Only a root user can run this command.

Examples

To display the update instructions for the **snaserv** software product on **/dev/rfd0**, enter:

```
inudocm snaserv
```

Files

/usr/sbin/inudocm	Contains the **inudocm** command.
/usr/lpp/*ProductName***/lpp.instr**	Specifies the update instructions for the software product.
/usr/lpp/*ProductName***/lpp.README**	Specifies special instructions for the software product.
/usr/lpp/*ProductName***/README**	Specifies special instructions for the software product.
/usr/lpp/*ProductName***/lpp.doc**	Specifies the updates to the documentation for the software product.

Related Information

The **installp** command, **restore** command.

---

inulag Command

Purpose

Acts as the front end to the subroutines to manage license agreements.

Syntax

inulag -r [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product*] [**-d** *Description* [**-m** *MessageSpecification*]] **-f** *File*

inulag -l | **-q** [**-c** | **-v**] [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product* | **-a**]

inulag -u [**-n** *FilesetName* | **-s** *FileName* | **-p** *Product*]

inulag -A

inulag -D

Description

The **inulag** command manages software license agreements. The basic forms are license agreement registration, license agreement listing, license agreement deactivation, license agreement validation, and license agreement revalidation.

The **-r** flag manages software license agreement registration of a fileset installed with **installp** or an independently-installed product installed through another installer. The path to a file that is always installed with an independently-installed product must be specified with the **-s** flag when the license agreement is registered.

The **-l** flag lists software license agreement registrations. If the **-c** flag is specified, the path to the software license agreement file is displayed rather than the contents of the file.

The **-q** flag queries for existence of software license agreements. A return code of 0 is returned if a license agreement exists. If the **-a** flag is also specified, then a return code of 0 is returned if there is a pending license agreement.

The **-u** flag removes the listing of software license agreements for a fileset or independently-installed product.

The **-D** flag forces revalidation of software license agreements upon the next system reboot.

Flags

-a	Used with the **-l** flag to show products that have a pending license agreement.
-A	Registers agreements for all pending license agreements.
-c	Used with the **-l** flag for colon-separated listing. Cannot be used with the **-v** flag.
-d *Description*	Specifies the default description for the fileset or product to which license applies.
-D	Forces the revalidation all license agreements on the next reboot.
-f *File*	Specifies the pathname specification for the license agreement. A '%L' in the specification is a substitution pattern for the current locale. **en_US** is the default locale. A ″%l″ in the specification matches the first two characters of the locale unless the current locale is **zh_CN**, in which all five characters of the locale designation are used.
-l	Lists software license agreements.
-m *MessageSpecification*	Specifies the message catalog for a translated description of the form ″catalog,set number,message number″.
-n *FilesetName*	Specifies the name of a fileset registered in the software vital product database governed by the license agreement.
-p *Product*	Specifies the product id, a nontranslatable alphanumeric string that uniquely identifies a product.
-q	Queries for license agreements. Does not show output. The value of 0 is returned if a license agreement exists. The **-q** flag can be used with other flags to query for particular license agreements or pending license agreements.
-r	Registers a software license agreement. Requires the **-f** flag for the path to the agreement file and either the **-n** flag or the **-s** flag to indicate the fileset name or signature file containing software subject to the agreement. The **-r** flag cannot be used with the **-l**, **-q**, or the **-u** flag. License agreements are registered as pending (status='P') during system installation, and NIM SPOT installation unless the environment variable **ACCEPT_LICENSES** is set to yes.
-s *FileName*	Specifies a signature file unique to installed software that identifies software not registered in the software vital product database that is governed by the license agreement. This is for use by software products not registering into the software vital product database. This form exists for the purpose of identifying software installed but not registered in the software vital product database. The *FileName* includes the full path to the file.

-u	Removes a license agreement. This does not actually remove the license agreement file, rather it changes the status of a license agreement associated with a fileset to inactive. Inactive license agreements do not need to be reagreed to, but they do not show up when listing installed software licenses.
-v	Used with the **-l** flag for verbose listing. Cannot be used with the **-c** flag.

Security

The agreement database is writable only by root. As a result, all flags other than the **-l** flag can only be used by a user operating with root user authority.

Related Information

The **installp** command, **lslpp** command, **nim** command.

Optional Software Products and Service Updates in *AIX 5L Version 5.2 Installation Guide and Reference*

---

inurecv Command

Purpose

Recovers files saved by the **inusave** command.

Syntax

inurecv *ProductName* [*OptionList*]

Description

The **inurecv** command recovers files and archive constituent files saved from a previous **inusave** command. It uses the **update.list** and **archive.list** files from the directory specified by the **INUSAVEDIR** environment variable. The **inurecv** command recovers files saved by program-provided installation or update procedures.

The **inurecv** command is primarily called by the **installp -r** command and the **installp -C** command to recover the files for a rejected program or a program that needs to be cleaned up.

The **inurecv** command is used to recover all the files for an installable program by separate calls to **inurecv** for the root, **/usr**, and **/usr/share** file trees. The save directories for the root, **/usr**, and **/usr/share** parts of an installation are:

- **/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**,
- **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** , and
- **/usr/share/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**

respectively, when set up by the **installp** command. *Level* refers to the level of the software product and has the format of *vv.rr.mmmm.ffff.ppppppppp*, where *vv* = version, *rr* = release, *mmmm* = modification, *ffff* = fix, and *ppppppppp* = fix ID (only for Version 3.2 images).

Parameters

OptionList	Specifies the full path name of a stanza file that contains the names of the separately installable options, such as **bosnet.tcp.obj**, that are to be recovered for the *ProductName* software product. The option names in the *OptionList* file must be specified one per line.

ProductName Specifies the installable software product, such as **bosnet**, whose files are to be recovered.

Environment Variables

INUEXPAND This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the recover (that is, the **-X** flag was passed to **installp**). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.

INUSAVE This flag is set to 1 by the **installp** command if files are to be saved (that is, the **-N** flag was not passed), and otherwise set to 0. The **inurecv** command attempts to recover files if **INUSAVE** is set to 1. If **INUSAVE** is set to 0, **inurecv** performs no recovery and exits with a return code of **INUGOOD**. If this environment variable is not set, the default is to attempt to recover files.

INUSAVEDIR The full path name to the directory where files are saved. If this environment variable is not set, then the directory used is **/usr/lpp/***ProductName***/inst_updt.save**.

ODMDIR The Object Data Manager object repository where the software vital product data is saved. If this environment variable is not set, the default directory used is **/etc/objrepos**.

Error Codes

INUBADC1 A copy of a file from one directory to another was unsuccessful.
INUGOOD No error conditions occurred.
INUNORP1 Unsuccessful replacement of a file in an archive file during program recovery.
INUNOSAV The save directory does not exist.
INUNOSVF A file that was saved in the save directory was not found.

Security

Privilege Control: Only the root user can run this command.

Examples

To recover all files previously saved for the **snaserv** program, enter:

```
inurecv snaserv
```

Files

/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Files saved for the root file tree.

/usr/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Files saved for the **/usr** file tree.

/usr/share/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Files saved for the **/usr/share** file tree.

Related Information

The **installp** command, **inusave** command.

---

inurest Command

Purpose

Performs simple archive and restore operations for the **installp** command and shell scripts. This command is used by the **installp** command and the install scripts.

Syntax

inurest [**-d** *Device*] [**-q**] *ListFile ProductName*

Description

The **inurest** command restores or archives all files listed in the file specified by the *ListFile* parameter.

If files are to be archived, there must be an archive control file, **/usr/lpp/***ProductName***/lpp.acf**, which contains entries in the following form:

*ComponentFile LibraryFile***.a.**

If the archive control file exists, the **inurest** command compares each of the file names in the *ListFile* file to the component files listed in **/usr/lpp/***ProductName***/lpp.acf**. Whenever the **inurest** command finds a match, the file name is added to a list of files that are archived. This list is then used to archive the restored files into a copy of the corresponding archive. When the archive is finished, the copy replaces the original file.

The *ListFile* parameter specifies the full path name of a file containing the relative path names, one per line, of files that a product needs to have restored.

The *ProductName* parameter specifies the software product to be restored.

Flags

-d *Device*	Specifies the input device. The default device is the **/dev/rfd0** device.
-q	Specifies quiet mode. Suppresses the prompt from **restore**.

Environment Variables

INUEXPAND	This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the restore (that is, the **-X** flag was passed). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.
INULIBDIR	This is the directory where files that are specific to software product installation reside. If **INULIBDIR** is not set the **/usr/lpp/***ProductName* directory is used.
INUTEMPDIR	The directory to use for temporary space that is needed during the execution of this command. If this environment variable is not set, then the directory used is **/tmp**.

Error Codes

INUBADRC	Restoration of an updated version of files was unsuccessful.
INUBADMN	Unusable flag was specified.
INUCHDIR	Cannot change directory.
INUGOOD	No error conditions occurred.
INUNOAP2	Unable to access the apply list.
INUNORP2	Failed replacing a constituent file in the archive file.
INUTOOFW	One or more parameters are missing.
INUTOOMN	Too many parameters are specified.

Security

Privilege Control: Only the root user can run this command.

Examples

To restore all the files listed in the **ac** file for the **snaserv** program, enter:

```
inurest /usr/lpp/snaserv/ac snaserv
```

Files

$INULIBDIR/lpp.acf	Archive control file.

Related Information

The **installp** command, **inucp** command, **inurecv** command, **inusave** command.

---

inurid Command

Purpose

Removes information used for installation of diskless/dataless clients from the **inst_root** directories of installed software.

Syntax

inurid [**-q** | **-r**]

Description

The **inurid** command is used to remove files stored in the **inst_root** directories of installed software.

The names of these directories are of the forms: **/usr/lpp/***PackageName***/inst_root** for software products and **/usr/lpp/***PackageName***/***OptionName***/v.r.m.f/inst_root** for AIX Version 4 updates.

When this command is called, the **inst_root** directories are removed for all products and updates in the committed state. Also, an indicator is stored in the Software Vital Product Data indicating that the proper **inst_root** directory information is to be removed after the completion of each future installation action, for example, actions performed by the **installp** command.

> **Attention:** One reason a user may want to remove **inst_root** directories is to save disk space. The implication of removing these directories is that the system cannot be used as a Shared Product Object Tree (SPOT) server of diskless/dataless clients. Also, once **inst_root** directories are removed from a system, there is no way to retrieve the directories. Therefore, the system cannot later be converted to a SPOT server without reinstalling the entire operating system.

Flags

-q Queries whether **inst_root** directories have been removed from the system. A return value of 0 indicates that **inst_root** directories have not been removed and a return value of 1 indicates that the **inst_root** directories have been removed.

-r Requests **inst_root** directories be removed from the system.

Security

Privilege Control: You must be the root user to run this command.

Files

/usr/lib/instl/inurid	Contains the **inurid** command.

Related Information

The **installp** command.

---

inusave Command

Purpose

Saves files that are installed or updated during an installation procedure. This command is used by the **installp** command and the install scripts.

Syntax

inusave *ListFile ProductName*

Description

The **inusave** command saves the files and archived files that are listed in the file specified by the *ListFile* parameter for the *ProductName* software product. The **inusave** command is designed for use with the **installp** command.

The **inusave** command creates the **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** directory if it does not already exist, where *Level* has the form *vv.rr.mmmm.ffff* and *vv* = the version, *rr* = the release, *mmmm* = the modification, and *ffff* = fix. This is the directory in which the installation procedures store saved files. The save directory is defined by the **INUSAVEDIR** environment variable.

The save directories for the / (root), **/usr**, and **/usr/share** parts of an installation are:

- **/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**,
- **/usr/lpp/***PackageName***/***FilesetName***/V.R.M.F.save** , and
- **/usr/share/lpp/***PackageName***/***FilesetName***/V.R.M.F.save**

respectively, when set up by the **installp** command. The **installp** command calls **inusave** for each of these three directories. The *ListFile* parameter is the full path name of the file that lists the files that are to be saved if a current copy exists.

If a file named in the *ListFile* file already exists, the **inusave** command copies that file to the **$INUSAVEDIR/update**.*n* file, where *n* is an integer assigned by the **inusave** command. If the file does not exist, the **inusave** command assumes that this entry in the *ListFile* parameter represents either a new file or a file to be archived or processed by the archive procedure described later in this section.

The **inusave** command maintains a list of saved files in the **$INUSAVEDIR/update.list** file. This file is a stanza file with an entry for each saved file. Entries in the **update.list** file resemble the following:

```
/usr/bin/chkey:
         update.n = update.1
         option = bosnet.nfs.obj
         _id = 209
         _reserved = 0
         _scratch = 0
         lpp_id = 72
         private = 0
         file_type = 0
         format = 1
         loc0 = /usr/bin/chkey
```

```
        size = 7800
        checksum = 44561

/usr/bin/domainname:
        update.n = update.2
        option = bosnet.nfs.obj
        _id = 210
        _reserved = 0
        _scratch = 0
        lpp_id = 72
        private = 0
        file_type = 0
        format = 1
        loc0 = /usr/bin/domainname
        size = 2526
        checksum = 12439
```

In the previous example **/usr/bin/chkey** (the name of the stanza) is the name of an original file that was saved and **update.1** is the name of the file in the **$INUSAVEDIR** directory to which it was copied. The file **/usr/bin/chkey** belongs to the **bosnet.nfs.obj** installable option of the software product **bosnet**. The stanza name and the first two items in the stanza (update.n and option) exist for each stanza in the **update.list** file. The remaining items in the stanza, which may vary, are information from the Software Vital Product Data (SWVPD) database.

An archived constituent file is saved if there is a valid archive control file, **lpp.acf**, in the current directory. If the **lpp.acf** file exists, the **inusave** command compares each of the file names in *ListFile* to the constituent file names in **lpp.acf**. When it finds a match, the **inusave** command uses the **ar** command to extract the constituent file from its associated archive file. It then moves the file to the **$INUSAVEDIR/archive**.*n* file, where *n* is an integer selected by the **inusave** command.

The **inusave** command maintains a list of the extracted files that have been saved in the **$INUSAVEDIR/archive.list** file. This file is a stanza file with an entry for each saved constituent file. Entries in the **archive.list** file resemble the following:

```
/prodx.filea:
        archive.n = archive.1
        arc_name = /usr/lib/productx/libprodx.a
        option = productx.option1.obj
        _id = 833
        _reserved = 0
        _scratch = 0
        lpp_id = 7
        private = 0
        file_type = 0
        format = 1
        loc0 = /prodx.filea
        loc1 = "h11,h12"
        loc2 =
"/usr/lpp/productx.filea/s11,/usr/lpp/productx.filea/s12"
        size = 1611
        checksum = 62793
```

In the previous example **/prodx.filea** (the name of the stanza) is the name of the original constituent file that was saved and **archive.1** is the name of the file in the **$INUSAVEDIR** directory to which it was copied. The **/usr/lib/productx/libprodx.a** is the full path name of the archive file defined in the **lpp.acf** archive control file. The constituent file **/prodx.filea** belongs to the **productx.option1.obj** installable option of the software product **productx**. The stanza name and the first three items in the stanza (archive.n, arc_name, and option) will exist for each stanza in the **archive.list** file. The remaining items in the stanza, which may vary, are information from the SWVPD database.

Parameters

ListFile Specifies the full path name of the file containing a list of relative path names, one per line, of files that are to be saved.

ProductName Specifies the installable software product whose files are to be saved.

Environment Variables

INUEXPAND This flag is set to 1 by the **installp** command if file systems are to be expanded if necessary to do the save (that is, the **-X** flag was passed to **installp**). It is set to 0 if file systems are not to be expanded. If this environment variable is not set, the default is not to expand file systems.

INUSAVE This flag is set to 1 by the **installp** command if files are to be saved (that is, the **-N** flag was not passed to **installp**). It is set to 0 if files are not to be saved. If this environment variable is not set, the default is to save files.

INUSAVEDIR The full path name to the directory where files are to be saved. If this environment variable is not set, then the directory to be used is **/usr/lpp/***ProductName***/inst_updt.save**.

INUTEMPDIR The directory to use for temporary space that is needed during the execution of this command. If this environment variable is not set, then the directory used is **/tmp**.

Error Codes

The following error codes are defined in **/usr/include/inuerr.h**:

INUBADSC A save directory could not be created.

INUBADC2 A file could not be copied from one directory to another.

INUGOOD No error conditions occurred.

INUNOAP1 Could not access *ListFile*.

INUTOOFW One or more parameters were missing.

INUTOOMN Too many parameters were specified.

Security

Privilege Control: Only the root user can run this command.

Examples

To save all the files listed in the **snaserv.al** file of the **snaserv** program, enter:

```
inusave /usr/lpp/snaserv/snaserv.al snaserv
```

Files

/usr/lpp/*PackageName***/lpp.acf**
Specifies the archive control file.

/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Specifies the save directory for the root.

/usr/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Specifies the save directory for the **/usr** files.

/usr/share/lpp/*PackageName***/***FilesetName***/V.R.M.F.save**
Specifies the save directory for the **/usr/share** files.

Related Information

The **installp** command, **inurecv** command.

inutoc Command

Purpose

Creates a **.toc** file for directories that have backup format file install images. This command is used by the **installp** command and the install scripts.

Syntax

inutoc [*Directory*]

Description

The **inutoc** command creates the **.toc** file in *Directory*. If a **.toc** file already exists, it is recreated with new information. The default installation image *Directory* is **/usr/sys/inst.images**. The **inutoc** command adds table of contents entries in the **.toc** file for every installation image in *Directory*.

The **installp** command and the **bffcreate** command call this command automatically upon the creation or use of an installation image in a directory without a **.toc** file.

Error Codes

INUBADIR	Usage error or *Directory* did not specify a directory.
INUCHDIR	Unable to change directories to *Directory*.
INUCRTOC	Could not create the **.toc** file.
INUGOOD	No errors occurred.
INUSYSFL	A system call failed.

Security

Privilege Control: Only the root user can run this command.

Examples

1. To create the **.toc** file for the **/usr/sys/inst.images** directory, enter:
   ```
   inutoc
   ```
2. To create a **.toc** file for the **/tmp/images** directory, enter:
   ```
   inutoc /tmp/images
   ```

Files

/usr/sys/inst.images	The default directory to create a **.toc** file.
.toc	The file created by this command in the specified directory.

Related Information

The **bffcreate** command, **installp** command.

inuumsg Command

Purpose

Displays specific error or diagnostic messages provided by a software product's installation procedures. This command is used by the **installp** command and the install scripts.

Syntax

inuumsg *Number* [*Argument1*] [**,** *Argument2*] [**,** *Argument3*] [**,** *Argument4*]

Description

The **inuumsg** command displays error or diagnostic messages for a software product's installation procedures. Rather than each procedure having its own text, messages are maintained in a central message catalog, **/usr/lpp/msg/$LANG/inuumsg.cat**. When you run the **inuumsg** command and specify the message *Number*, the error message is displayed. Up to four string arguments, *Argument1* to *Argument4*, can be substituted into the message in the appropriate location.

Return Values

0 Indicates the message was found and displayed.
1 Indicates the message was not found and not displayed.

Security

Privilege Control: Only the root user can run this command.

Examples

To see error message number 3, enter:

```
inuumsg 3
```

Files

/usr/lpp/msg/$LANG/inuumsg.cat	The message catalog.

Related Information

The **installp** command.

---

invscout Command

Purpose

Surveys the host system for currently installed microcode or Vital Product Data (VPD).

Syntax

invscout [**-v**] [**-c**] [**-r**] [**-m** *machine_type_and_model*] [**-s** *serial_number*] [*-catl microcode_catalog_path*][**-g**] [**-q**] [**-h**]

Description

The **invscout** command executes one instance of the stand-alone version of the Inventory Scout process. The **invscoutd** command starts the server daemon side of a client-server version.

The Inventory Scout process supports two survey types:

- Microcode Survey
- Vital Product Data (VPD) Survey (**-v**)

Microcode Survey

A Microcode Survey gathers data from the host system on currently installed microcode for **invscout**-supported systems, devices and adapters. It compares the gathered microcode levels to the latest levels available, and stores the data in a **Microcode Survey Upload File** that can be displayed locally via the webSM GUI, or uploaded to a Web server via the Internet.

A Microcode Survey also produces a **Microcode Survey Formatted Text Report File**. This file can be printed or displayed on a monitor and contains a subset of the information recorded in the upload file. This subset includes information about the **invscout** execution itself and the levels of currently installed microcode. The **-r** flag causes this report also to be sent to the screen from where the command was invoked.

All of the previous reports can contain information on the following:

- system microcode
- service microcode
- device and adapter microcode

VPD Survey (-v)

A VPD Survey stores the system VPD in a **VPD Survey Upload File** that can be uploaded to a Web server via the Internet . Once on a Web server, a CGI forwards the file to a repository and produces a Web page indicating the status of the operation.

No formatted text report is available for VPD Surveys.

Survey Results Concatenation (-c)

This option concatenates two or more **Microcode Survey Upload Files** into a single **Microcode Survey Concatenated Upload File** or two or more VPD Survey Upload Files into a single **VPD Survey Concatenated Upload File**. A Concatenated Upload File can be uploaded to a Web server using the Internet and processed by the server CGI to give the same results as would have been obtained by uploading and processing all the component files individually. The input files can be any valid upload files but, typically, this operation is done to simplify the task of uploading the results from several host systems.

- The version of the command executing the concatenation and the versions of the commands that produced the files to be concatenated must all be the same.
- Microcode Survey Upload Files cannot be concatenated with VPD Survey Upload Files.
- Versions 2.1.0.0 and subsequent versions of this command do not require concatenation of Microcode Survey Upload Files, because the files are processed locally.

To concatenate a set of existing Microcode Survey upload files, do the following:

1. Copy the files into the **Microcode Survey Concatenation Input Directory**.
2. Execute:

   ```
   invscout -c
   ```

3. Find the output **Microcode Survey Concatenated Upload File** in the same directory as the upload file for a Microcode Survey.

To concatenate a set of existing VPD Survey upload files, do the following:

1. Copy the files into the **VPD Survey Concatenation Input Directory**.
2. Execute:

   ```
   invscout -v -c
   ```

3. Find the output **VPD Survey Concatenated Upload File** in the same directory as the upload file for a VPD Survey.

Flags

-v	Sets the survey or concatenation type to VPD (the default is Microcode).
-c	Concatenates existing survey upload files (the default is to perform a new survey).
-r	For a Microcode Survey, sends a copy of the formatted text report file to the screen from which the command was invoked. This flag is ignored if either the **-v** or the **-c** flag is used.
-m *machine_type_and_model*	For a VPD survey, allows input of the host platform machine type and model for hosts that use/require this information.
-s *serial_number*	For a VPD survey, allows input of the host serial number for hosts that use/require this information.
-catl microcode_catalog_path	Overrides the default location of the microcode catalog path.
-g	Displays the versions of this command and of the logic database currently in use.
-q	Suppresses most run-time messages.
-h	Generates a help (usage) statement. If this flag is used, all other flags are ignored.

Exit Status

This command returns the following exit values:

0	Indicates successful completion.
Non-zero	Indicates an error occurred.

If an error occurs, the command writes an **error log**.

Security

This command is owned by root, and is installed with the **setuid** bit ON so that any user can run it.

Examples

1. To run one Microcode Survey and send the results to a formatted text report file and an upload file, type:

   ```
   invscout
   ```

2. To run one VPD Survey and send the results to an upload file, type:

   ```
   invscout -v
   ```

3. To concatenate previously produced Microcode Survey upload files into a single upload file, type:

   ```
   invscout -c
   ```

   Note: Only applicable to Versions of this command prior to 2.1.0.0.

4. To concatenate previously produced VPD Survey upload files into a single upload file, type:

   ```
   invscout -v -c
   ```

Files

/usr/sbin/invscout	Contains the **invscout** command.
/var/adm/invscout/*host*.mup	Microcode Survey Upload File. The *host* variable is the host name of the system represented in the file.
/var/adm/invscout/invs.mrp	Microcode Survey Formatted Text Report File.
/var/adm/invscout/*host*.vup	VPD Survey Upload File. The *host* variable is the host name of the system represented in the file.
/var/adm/invscout/invs.mic.con.inp	Microcode Survey Concatenation Input Directory.

/var/adm/invscout/invs.vpd.con.inp	VPD Survey Concatenation Input Directory
/var/adm/invscout/invs.con.mup	Microcode Survey Concatenated Upload File.
/var/adm/invscout/invs.con.vup	VPD Survey Concatenated Upload File.
/var/adm/invscout/invs.err	Error log written if the command encounters an error.
/var/adm/invscout/microcode	Directory for microcode-related actions. Default location for microcode catalog file.
/var/adm/invscout/microcode/catalog.mic	Default microcode catalog file.
/var/adm/invscout/invscout.log	Log file.
/tmp/invscout	Holds **invscout** temporary files. All files in this directory are deleted at the start of every execution of this command.

Related Information

The **invscoutd** command.

---

invscoutd Command

Purpose

Launches a permanent Inventory Scout server daemon.

Syntax

invscoutd [**-o**] [**-p** *Portno*] [**-b** *Bufsize*] [**-t** *Timeout*] [**-v** *Verblev*]

Description

The **invscoutd** command implements a permanent Inventory Scout server daemon on one machine in a user's local network. The usual client is a Java applet running in the user's Web browser, which was downloaded from a central Inventory Scout CGI application.

Daemon initialization involves reading command line options and several local Inventory Scout companion files. When in operation, each client-server transaction involves reading from a well-known socket for a text string and returning a text report over the same socket.

The daemon maintains a record of its actions in a log file. Depending on the specified verbosity level, the log lines may contain startup and shutdown banners, traces of each call, detailed internal program traces, and error statements. Depending on the specified verbosity level, startup banners may also be written to **stderr**.

Protocols

Client connections to the daemon's socket use the Internet TCP/IP protocol. In a transaction, the invoking client applet sends an action request, as a URL-encoded text string, to the server daemon. The request is by any ASCII control character (x00 to x1F), which triggers the processing of the request.

Some requests require the client to pass additional data. In these cases, the additional data immediately follow the termination byte for a length specified in the action request.

With one exception (ACTION=PING), the server daemon always returns a pseudo MIME format text report written back over the same socket connection. The pseudo MIME format is used even for error results. The daemon terminates the returned text and the transaction itself by closing the socket, resulting in an end-of-file (EOF) indication to the invoking client. The client should close the socket at its end of the connection as soon as the EOF is received.

URL-encoded message

The action request string is a standard URL-encoded string. For example:

```
"ACTION=actionword&NAME1=value1&NAME2&NAME3=word%xx+word+word\0"
```

Supported Field Names and Values

Name	Meaning/Use	Supported Values
ACTION	See the action request table that follows.	The left-hand column of the action request table constitutes a list of supported Values.
MRDM	Allows the client to provide a (cleartext) password for any ACTION that uses/requires this information. The value is case sensitive.	Any ASCII string (case sensitive).
DATALEN	This name must be present if additional binary data immediately follow an ACTION string termination byte, and must be absent if no additional data follow the termination byte. The integer value provided specifies the number of additional data bytes. If the client attempts to write more data than this, if the action does not accept the DATALEN parameter and discards any additional data, or if the action processor detects an early error, the daemon may prematurely close the client-to-server socket pipe. A transaction with n greater than a specific maximum value will immediately return an error code (see the **-d** command line option).	Any integer up to the value implied by the presence or absence of the **-d** command line option
CLIENT	Allows the client to identify itself for any ACTION that uses/requires this information.	The HSC value instructs Inventory Scout to allow certain actions that are only allowed when under the control of an HMC Inventory Scout master.
MODEL	Allows the client to inform the server of the server's model number for VPD surveys that use/require this information.	Any ASCII string of up to 25 characters (restrictions apply with some machines)
SERIAL	Allows the client to inform the server of the server's serial number for VPD surveys that use/require this information.	Any ASCII string of up to 25 characters (restrictions apply with some machines)

Notes:

1. Field names and their values are separated by equal signs (=).
2. **Name**=*Value* pairs are separated by an & character.
3. The **Name** field is always case insensitive.
4. The *Value* field is case insensitive, unless documented otherwise.
5. The **ACTION**=*keyword* pair must always be present.
6. A string between ampersands without an equal sign is parsed as a **Name** with an Empty value.
7. Spaces can be represented by + (plus signs).
8. Binary characters may be coded as the escape sequence of a percent sign followed by exactly two hexadecimal chars (%xx). This escape sequence must also be used to code URL metacharacters like the &, = (equal sign), and + (plus sign) within a Value.
9. The control character termination byte must always be sent by the client.

Action Requests

Action	MRDM	Description

PING	not required	The daemon *immediately* closes the socket, causing an immediate EOF in the client. This is the only action that does not return a result code or text of any kind. Example:
		`"action=ping\0"`
		
		`<EOF>`

ECHO	not required	The daemon returns a text report consisting of the original unparsed request string followed by a linefeed. A password (MRDM) is not required but will be echoed if provided, along with everything else. Additional data (DATALEN) are not required but will be echoed if present, as is, after the request string. For the ECHO request, DATALEN will be silently truncated to a maximum of 2000. Example:
		`"action=ECHO&MRDM=xyz&datalen=5\0abcde"`
		
		`"RESULT=0\n"`
		`"\n"`
		`"action=ECHO&MRDM=xyz&datalen=5\n"`
		`"abcde"<EOF>`

URLDECODE	not required	The daemon returns a text report of the request string after parsing, and an exact copy of any subsequent data. A password (MRDM) is not required but will be parsed and returned if provided. Additional data (DATALEN) are not required but will be parsed and returned if provided; however, any actual additional data beyond the request string will be discarded. Each numbered line of the report exhibits one parsed Name=Value pair from the original string. Example:
		`"action=UrlDecode&subaction=xyz\0"`
		
		`"RESULT=0\n"`
		`"\n"`
		`"  0: ACTION       UrlDecode\n"`
		`"  1: SUBACTION    xyz\n"`
		`<EOF>`

TESTPWD	required	The daemon returns RESULT=0 if the MRDM password is valid. Otherwise it returns RESULT=2. Additional data (DATALEN) are not accepted and will be discarded if present. Example:
		`"ACTION=TESTPWD&MRDM=thepassword\0"`
		
		`"RESULT=0\n"`
		`"\n"`
		`<EOF>`

VERSIONS	not required	The daemon reports the current version numbers of the Inventory Scout itself. Additional data (DATALEN) are not accepted and will be discarded if present. Example:
		`"ACTION=VERSIONS\0"`
		
		`"RESULT=0\n"`
		`"\n"`
		`"1.2.3.4\n"`
		`"5.6.7.8\n"`
		`<EOF>`

Action Requests

Action	MRDM	Description

CATALOG	required	The daemon updates the scout's microcode catalog file with the file data passed. Both a password and the data length parameter must be included in the request string. The daemon does not necessarily have to execute as root for this action but it must have file write permissions to **/var/adm/invscout/microcode/catalog.mic**. Example:
		`"ACTION=CATALOG&MRDM=xyz&DATALEN=17042\0"`
		`"...17042 bytes of ascii data..."`
		
		`"RESULT=0\n"`
		`"\n"`
		`<EOF>`

MCODES	required	The daemon executes the Microcode Survey Option. Additional data (DATALEN) are not accepted and will be discarded if present. Example:
		`"ACTION=MCODES&MRDM=xyz\0"`
		
		`"RESULT=0\n"`
		`"\n"`
		`"Report Line 1\n"`
		`"Report Line 2\n"`
		`:`
		`:`
		`"Report Line N\n"`
		`<EOF>`

VPDS	required	The daemon executes the VPD Survey Option. Additional data (DATALEN) are not accepted and will be discarded if present. Example:
		`"ACTION=VPDS&MRDM=xyz\0"`
		
		`"RESULT=0\n"`
		`"\n"`
		`"Report Line 1\n"`
		`"Report Line 2\n"`
		`:`
		`:`
		`"Report Line N\n"`
		`<EOF>`

Results

The daemon returns a text result in a pseudo MIME format. It returns a header consisting of one or more **Name**=*Value* pairs, each on a line by itself. The first **Name**=*Value* pair always is the result code in the form **RESULT**=*number*. The result code always is returned for every action, except the PING action.

Internal scout result codes applicable only to the Java applet client are not documented in the following information.

An optional free-form text report may follow the header lines depending on the result code. If there is a free-form text report, the header is first terminated by an empty line, such as two adjacent linefeeds.

In any event, the result report is terminated by an EOF indicator after reading the last of the report text from the socket. The EOF also signifies the end of the transaction itself.

Result Codes

Result=	Description
0	Complete success.
1	Daemon aborted due to memory allocation error. This can happen in either the parent server daemon or one of the service children.

Result Codes

Result=	Description
2	Service child daemon aborted because the required password (MRDM=password) was missing or not valid.
3	Service child daemon aborted because the action name-value pair (ACTION=keyword) was missing or not valid.
4	Service child daemon aborted because it was unable to reset its user ID to `invscout`.
21	Service child daemon aborted due to overflow of socket input buffer. The text report part of the result is a native language error message. Client must reduce the length of the request string, or kill and restart the daemon with an increased buffer size.
22	Service child daemon aborted due to socket read error. The text report part of the result is a native language error message including the system's I/O errno string. A logfile entry will also contain the system's errno string.
23	Service child daemon aborted due to socket read timeout. The text report part of the result is a native language error message. Client must send a control character termination byte after the end of the request string, and must always send as many data bytes as specified in the DATALEN parameter. The timeout period may be changed with the -t command line argument.
24	Service child daemon aborted due to premature EOF while reading request string. The text report part of the result is a native language error message. Client must send a termination byte after the end of the request string before closing the socket connection.
25	Service child daemon aborted due to missing or invalid DATALEN parameter for an action that requires it. The text report pair of the result is a native language error message. Client must send the length of the data for all actions which pass additional binary data beyond the URL-encoded request string. Most such actions also require that the DATALEN value be limited to a specific maximum size.
26	Service child daemon aborted due to regular file I/O error, such as a permissions error, out of disk space, and so on. The text report part of the result is a native language error message. Usually, the I/O problem must be corrected on the server machine before the client can attempt the action again.
27	Service child daemon aborted because it was unable to retrieve the version number for an activity that required it.

Flags

Specify any arguments, beginning with a hyphen (-). Space is not allowed between a flag and its value.

-o Overwrites an existing logfile. If the **-o** flag is not specified, new logfile lines are appended to any existing logfile.

-p *Portno* Changes this server's port number from the default 808 to *Port*.

-b *Bufsize* Inventory Scout commands are specified as URL-encoded strings read from a TCP/IP socket into a 1024 byte fixed length buffer. The **-b** flag can change the buffer size to *Bufsize* bytes if future protocol changes require a larger read buffer.

-t *Timeout* The client applet writes a control character termination byte at the end of the URL-encoded request string to indicate the end of the request. If the **invscoutd** daemon does not receive the termination byte within a timeout period, it aborts the transaction and closes the socket. Similarly the client must send all bytes of the additional data specified in the DATALEN parameter with sufficient speed to prevent timeout between read blocks. The **-t** option changes the default timeout period from 30 seconds to *Timeout* seconds.

-v *Verblev*

The amount of detail written to the logfile and stderr depends on the verbosity level of the daemon. Each level incorporates the messages in the lower levels; increasing the verbosity level increases the number and types of messages that are written. The verbosity level is an integer ranging from 0 to 25. The **-v** flag changes the verbosity level from the default 18 to *Verblev*.

Verbosity Levels

Level	Description
0	All error and status messages disabled.
5	Only fatal error messages are written. Fatal errors result in the death of the server. Usually, similar messages are written to both the *Logfile* and stderr.
10	All error messages are written. These include nonfatal errors such as protocol errors, as well as fatal errors. Nonfatal error messages are usually written only to the *Logfile*.
15	This level includes startup and shutdown banner messages. Simple banner messages are usually written to both the *Logfile* and stderr.
18	This level includes call trace status messages. Every client call results in a single trace message. This is the default level for the **invscoutd** daemon. Trace messages are written only to the *Logfile*.
20	This level includes program trace messages. Program traces are fairly detailed program execution status messages typically used for debugging purposes. This level is not suitable for usual production execution because over time, it floods the *Logfile* with large amounts of text. Trace messages are written only to the *Logfile*.
25	This is the maximum level and includes extensive program debug messages. This level is not suitable for usual production execution. Trace messages are written only to the *Logfile*.

Exit Status

This command returns the following exit values:

0 Indicates successful initialization
Non-zero Indicates unsuccessful initialization

Security

The daemon must execute as effective user ID `0` (root). It is owned by root, and is installed with the ″setuid″ bit ON so that any user can launch it. At certain execution points, however, service children of the daemon reset their user ID to the authentication user ID `invscout`. The daemon will not execute unless the user `invscout` has been created on the host system.

By default, an accompanying cleartext password is required from the client for most operations. If the client's password does not match the system password for the authentication user ID `invscout`, the action exits with a return code. The authentication user ID cannot be changed.

Files

/usr/sbin/invscoutd Contains the **invscoutd** command
/etc/security/password Host system password file
/var/adm/invscout/microcode Directory for microcode-related actions. Default location for microcode catalog file.
/var/adm/invscout/microcode/catalog.mic Default microcode catalog file.
/var/adm/invscout/invscout.log Log file

Related Information

The invscout command.

---

ioo Command

Purpose

Manages Input/Output tunable parameters.

Syntax

ioo [**-p** | **-r**] { **-o** *Tunable* [**=***NewValue*] }

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

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

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

ioo -?

ioo -h [*Tunable*]

ioo -L [*Tunable*]

ioo -x [*Tunable*]

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

Description

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

The **ioo** command configures Input/Output tuning parameters. This command sets or displays current or next boot values for all Input/Output 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.

If a process appears to be reading sequentially from a file, the values specified by the **minpgahead** parameter determine the number of pages to be read ahead when the condition is first detected. The value specified by the **maxpgahead** parameter sets the maximum number of pages that are read ahead, regardless of the number of preceding sequential reads.

The operating system allows tuning of the number of file system **bufstructs** (**numfsbuf**) and the amount of data processed by the write-behind algorithm (**numclust**).

Attention: Misuse of the **ioo** command can cause performance degradation or operating-system failure. Before experimenting with **ioo**, you should be thoroughly familiar with Performance Overview of the Virtual Memory Manager (VMM).

Flags

-h [*Tunable*] Displays help about the *Tunable* parameter if one is specified. Otherwise, displays the **ioo** command usage statement.

-a Displays 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 is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and has been changed from its default value, and **-r** is not used in combination, it is not changed but a warning displays.

-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 are not changed but a warning displays.

-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 is not changed but a warning displays.

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

-p Specifies that the changes apply to both the current and reboot values when used in combination with the **-o**, **-d** or **-D** flags. 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, their current value cannot be changed.

When used with **-a** or **-o** without specifying a new value, the values display 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 the **-o**, **-d** or **-D** flags. That is, turns on the updating of the **/etc/tunables/nextboot** file. If any parameter of type **Bosboot** is changed, the user is 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.

-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
------------------------------------------------------------------------------
minpgahead                2      2      2      0      4K     4KB pages        D
     maxpgahead
------------------------------------------------------------------------------
maxpgahead                8      8      8      0      4K     4KB pages        D
     minpgahead
------------------------------------------------------------------------------
pd_npages                 64K    64K    64K    1      512K   4KB pages        D
------------------------------------------------------------------------------
maxrandwrt                0      0      0      0      512K   4KB pages        D
------------------------------------------------------------------------------
numclust                  1      1      1      0             16KB/cluster     D
------------------------------------------------------------------------------
numfsbufs                 186    186    186                                   M
------------------------------------------------------------------------------
...
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) and C (for Connect)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

-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) and C (for Connect)
    dtunable = space separated list of dependent tunable parameters
```

-? Displays the **ioo** command usage statement.

Any change (with **-o**, **-d** or **-D**) to a parameter of type Mount will result in a message being displayed 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 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**, will result 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, will result in an error message.

Tunable Parameters Type

All the tunable parameters manipulated by the tuning commands (**no**, **nfso**, **vmo**, **ioo**, 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

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 **ioo** command only includes Static, Dynamic, Mount and Incremental types.

Compatibility Mode

When running in pre 5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys0**, see **Tuning Enhancements for AIX 5.2** in the *AIX 5L Version 5.2 Performance Management Guide*), 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 5L Version 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 *AIX 5L Version 5.2 Performance Tools Guide and Reference* for more information.

Tunable Parameters

hd_pbuf_cnt

Purpose:

Specifies the total number of pbufs that the LVM uses.

Values:

Default: For the 32-bit kernel, the default is (4 + number of physical disks with an open logical volume) * 64. For the 64-bit kernel, the default is (4 + number of physical disks with an open logical valume) * 128.

Range:

Type: Incremental

Diagnosis:

Value of **hd_pendqblked** is non-zero, which indicates that LVM had to wait for pbufs. To see value of **hd_pendqblked**, use the **vmstat -v** command.

Tuning:

Useful to increase if there are a substantial number of simultaneous I/Os and value of **hd_pendqblked** is non-zero.

Refer To:

Miscellaneous I/O Tuning Parameters

lvm_bufcnt

Purpose:

Specifies the number of LVM buffers for raw physical I/Os.

Values:

Default: 9

Range: 1 to 64

Type: Dynamic

Diagnosis:

Applications performing large writes to striped raw logical volumes are not obtaining the desired throughput rate.

Tuning:

LVM splits large raw I/Os into multiple buffers of 128 K a piece. Default value of 9 means that about 1 MB I/Os can be processed without waiting for more buffers. If a system is configured to have striped raw logical volumes and is doing writes greater than 1.125 MB, increasing this value may help the throughput of the application. If performing larger than 1 MB raw I/Os, it might be useful to increase this value.

Refer To:

Miscellaneous I/O Tuning Parameters

maxpgahead

Purpose:

Specifies the maximum number of pages to be read ahead when processing a sequentially accessed file.

Values:

Default: 8 (the default should be a power of two and should be greater than or equal to minpgahead)

Range: 0 to 4096

Type: Dynamic

Diagnosis:

Observe the elapsed execution time of critical sequential-I/O-dependent applications with the time command.

Tuning:

Because of limitations in the kernel, do not exceed 512 as the maximum value used. The difference between minfree and maxfree should always be equal to or greater than maxpgahead. If execution time decreases with higher maxpgahead, observe other applications to ensure that their performance has not deteriorated.

Refer To:

Sequential Read-Ahead

maxrandwrt

Purpose:

Specifies a threshold (in 4 KB pages) for random writes to accumulate in RAM before subsequent pages are flushed to disk by the write-behind algorithm. The random write-behind threshold is on a per-file basis.

Values:

Default: 0

Range: 0 to largest_file_size_in_pages

Type: Dynamic

Diagnosis:

vmstat -n shows page out and I/O wait peaks on regular intervals (usually when the sync daemon is writing pages to disk).

Tuning:

Useful to set this value to 1 or higher if too much I/O occurs when **syncd** runs. Default is to have random writes stay in RAM until a sync operation. Setting maxrandwrt ensures these writes get flushed to disk before the sync operation has to occur. However, this could degrade performance because the file is then being flushed each time. Tune this option to favor interactive response time over throughput. After the threshold is reached, all subsequent pages are then immediately flushed to disk. The pages up to the threshold value stay in RAM until a sync operation. A value of 0 disables random write-behind.

Refer To:

VMM Write-Behind

minpgahead

Purpose:

Specifies the number of pages with which sequential read-ahead starts.

Values:

Default: 2

Range: 0 to 4096 (should be a power of two)

Type: Dynamic

Diagnosis:

Observe the elapsed execution time of critical sequential-I/O-dependent applications with time command.

Tuning:

Useful to increase if there are lots of large sequential accesses. Observe other applications to ensure that their performance has not deteriorated. Value of 0 may be useful if I/O pattern is purely random.

Refer To:

Sequential Read-Ahead

numclust

Purpose:

Specifies the number of 16 k clusters processed by the sequential write-behind algorithm of the VMM.

Values:

Default: 1

Range: 0 to any positive integer

Type: Dynamic

Diagnosis:

N/A

Tuning:

Useful to increase if there is a need to keep more pages in RAM before scheduling them for I/O when the I/O pattern is sequential. May be appropriate to increase if striped logical volumes or disk arrays are being used.

Refer To:

VMM Write-Behind

numfsbufs

Purpose:

Specifies the number of file system bufstructs.

Values:

Default: 186 (value is dependent on the size of the bufstruct)

Range:

Type: Mount

Diagnosis:

N/A

Tuning:

If the VMM must wait for a free bufstruct, it puts the process on the VMM wait list before the start I/O is issued and will wake it up once a bufstruct has become available. May be appropriate to increase if striped logical volumes or disk arrays are being used.

Refer To:

Miscellaneous I/O Tuning Parameters

pd_npages

Purpose:

Specifies the number of pages that should be deleted in one chunk from RAM when a file is deleted.

Values:

Default: 65536

Range: 1 to largest filesize_in_pages

Type: Dynamic

Diagnosis:

Real-time applications that experience sluggish response time while files are being deleted.

Tuning:

Tuning this option is only useful for real-time applications. If real-time response is critical, adjusting this option may improve response time by spreading the removal of file pages from RAM more evenly over a workload.

Refer To:

Miscellaneous I/O Tuning Parameters

sync_release_ilock

Purpose:

If set, will cause a **sync**() to flush all I/O to a file without holding the i-node lock, and then use the i-node lock to do the commit.

Values:

Default: 0 (off)

Range 0 or 1

Type: Dynamic

Diagnosis:

I/O to a file is blocked when the **syncd** daemon is running.

Tuning:

Default value of 0 means that the i-node lock is held while all dirty pages of a file are flushed.

Refer To:

Tuning File Syncs

j2_maxPageReadAhead

Purpose:

Specifies the maximum number of pages to be read ahead when processing a sequentially accessed file on Enhanced JFS.

Values:

Default: 8

Range:

Type: Dynamic

Diagnosis:

N/A

Tuning:

The difference between minfree and maxfree should always be equal to or greater than j2_maxPageReadAhead. If run time decreases with higher a j2_maxPageReadAhead value, observe other applications to ensure that their performance has not deteriorated.

j2_maxRandomWrite

Purpose:

Specifies a threshold for random writes to accumulate in RAM before subsequent pages are flushed to disk by the Enhanced JFS's write-behind algorithm. The random write-behind threshold is on a per-file basis.

Values:

Default: 0

Range:

Type: Dynamic

Diagnosis:

N/A

Tuning:

Useful if too many pages are flushed out by **syncd**.

j2_minPageReadAhead

Purpose:

Specifies the minimum number of pages to be read ahead when processing a sequentially accessed file on Enhanced JFS.

Values:

Default: 2

Range:

Type: Dynamic

Diagnosis:

N/A

Tuning:

Useful to increase if there are lots of large sequential accesses. Observe other applications to ensure that their performance has not deteriorated. Value of 0 may be useful if I/O pattern is purely random.

j2_nBufferPerPagerDevice

Purpose:

Specifies the number of file system bufstructs for Enhanced JFS.

Values:

Default: 512

Range:

Type: Mount

Diagnosis:

Using **vmstat -v**, the value of xpagerbufwaitcnt increases fast.

Tuning:

If the kernel must wait for a free bufstruct, it puts the process on a wait list before the start I/O is issued and will wake it up once a bufstruct has become available. May be appropriate to increase if striped logical volumes or disk arrays are being used.

j2_nPagesPerWriteBehindCluster

Purpose:

Specifies the number of pages per cluster processed by Enhanced JFS's write behind algorithm.

Values:

Default: 32

Range:

Type: Dynamic

Diagnosis:

N/A

Tuning:

Useful to increase if there is a need to keep more pages in RAM before scheduling them for I/O when the I/O pattern is sequential. May be appropriate to increase if striped logical volumes or disk arrays are being used.

j2_nRandomCluster

Purpose:

Specifies the distance apart (in clusters) that writes have to exceed in order for them to be considered as random by the Enhanced JFS's random write behind algorithm.

Values:

Default: 0

Range:

Type: Dynamic

Diagnosis:

N/A

Tuning:

Useful to increase if there is a need to keep more pages in RAM before scheduling them for I/O when the I/O pattern is random and random write behind is enabled (j2_maxRandomWrite).

Examples

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

   ```
   ioo -L
   ```

2. To turn sync_release_ilock on, type:

   ```
   ioo -o sync_release_ilock=1
   ```

3. To display help on j2_nPagesPerWriteBehindCluster, type:

   ```
   ioo -h j2_nPagesPerWriteBehindCluster
   ```

4. To set maxrandwrt to 4 after the next reboot, type:

   ```
   ioo -r -o maxrandwrt=4
   ```

5. To permanently reset all ioo tunable parameters to default, type:

   ```
   ioo -p -D
   ```

6. To list the reboot value of all ioo parameters, type:

   ```
   ioo -r -a
   ```

7. To list (spreadsheet format) the current and reboot value, range, unit, type and dependencies of all tunables parameters managed by the **ioo** command, type:

   ```
   ioo -x
   ```

Related Information

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

Kernel Tuning in *AIX 5L Version 5.2 Performance Tools Guide and Reference*.

Tuning Enhancements for AIX 5.2 in *AIX 5L Version 5.2 Performance Management Guide*.

---

iostat Command

Purpose

Reports Central Processing Unit (CPU) statistics and input/output statistics for the entire system, adapters, tty devices, disks and CD-ROMs.

Syntax

iostat [**-s**] [**-a**] [**-d** | **-t**] [**-T**][**-m**][*PhysicalVolume* ...] [*Interval* [*Count*]]

Description

The **iostat** command is used for monitoring system input/output device loading by observing the time the physical disks are active in relation to their average transfer rates. The **iostat** command generates reports that can be used to change system configuration to better balance the input/output load between physical disks and adapters.

The first report generated by the **iostat** command provides statistics concerning the time since the system was booted. Each subsequent report covers the time since the previous report. All statistics are reported each time the **iostat** command is run. The report consists of a tty and CPU header row followed by a row of tty and CPU statistics. On multiprocessor systems, CPU statistics are calculated system-wide as averages among all processors.

If the **-s** flag is specified, a system-header row is displayed followed by a line of statistics for the entire system. The hostname of the system is printed in the system-header row.

If the **-a** flag is specified, an adapter-header row is displayed followed by a line of statistics for the adapter. This will be followed by a disk-header row and the statistics of all the disks/CD-ROMs connected to the adapter. Such reports are generated for all the disk adapters connected to the system.

A disks header row is displayed followed by a line of statistics for each disk that is configured. If the *PhysicalVolume* parameter is specified, only those names specified are displayed.

If the *PhysicalVolume* parameter is specified, one or more alphabetic or alphanumeric physical volumes can be specified. If the *PhysicalVolume* parameter is specified, the tty and CPU reports are displayed and the disk report contains statistics for the specified drives. If a specified logical drive name is not found, the report lists the specified name and displays the message `Drive Not Found`. If no Logical Drive Names are specified, the report contains statistics for all configured disks and CD-ROMs. If no drives are configured on the system, no disk report is generated. The first character in the *PhysicalVolume* parameter cannot be numeric.

The *Interval* parameter specifies the amount of time in seconds between each report. The first report contains statistics for the time since system startup (boot). Each subsequent report contains statistics collected during the interval since the previous report. The *Count* parameter can be specified in conjunction with the *Interval* parameter. If the *Count* parameter is specified, the value of count determines the number of reports generated at *Interval* seconds apart. If the *Interval* parameter is specified without the *Count* parameter, the **iostat** command generates reports continuously.

The **iostat** command is useful in determining whether a physical volume is becoming a performance bottleneck and if there is potential to improve the situation. The % utilization field for the physical volumes indicates how evenly the file activity is spread across the drives. A high % utilization on a physical volume is a good indication that there may be contention for this resource. Since the CPU utilization statistics are also available with the **iostat** report, the percentage of time the CPU is in I/O wait can be determined at the same time. Consider distributing data across drives if the I/O wait time is significant and the disk utilization is not evenly distributed across volumes.

> **Note:** Some system resource is consumed in maintaining disk I/O history for the **iostat** command. Use the **sysconfig** subroutine, or the System Management Interface Tool (SMIT) to stop history accounting.

Reports

The **iostat** command generates four types of reports, the tty and CPU Utilization report, the Disk Utilization report, the System throughput report and the Adapter throughput report.

tty and CPU Utilization Report: The first report generated by the **iostat** command is the tty and CPU Utilization Report. For multiprocessor systems, the CPU values are global averages among all processors. Also, the I/O wait state is defined system-wide and not per processor. The report has the following format:

Column	Description
`tin`	Shows the total number of characters read by the system for all ttys.
`tout`	Shows the total number of characters written by the system to all ttys.
`% user`	Shows the percentage of CPU utilization that occurred while executing at the user level (application).
`% sys`	Shows the percentage of CPU utilization that occurred while executing at the system level (kernel).
`% idle`	Shows the percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.
`% iowait`	Shows the percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.

This information is updated at regular intervals by the kernel (typically sixty times per second). The tty report provides a collective account of characters per second received from all terminals on the system as well as the collective count of characters output per second to all terminals on the system.

Methods Used to Compute CPU Disk I/O Wait Time: Operating system version 4.3.3 and later contain 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 AIX 4.3.3 and later is as follows: The change in operating system version 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.

Disk Utilization Report: The second report generated by the **iostat** command is the Disk Utilization Report. The disk report provides statistics on a per physical disk basis. The report has a format similar to the following:

`% tm_act`	Indicates the percentage of time the physical disk was active (bandwidth utilization for the drive).
`Kbps`	Indicates the amount of data transferred (read or written) to the drive in KB per second.
`tps`	Indicates 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 total number of KB read.
`Kb_wrtn`	The total number of KB written.

Statistics for CD-ROM devices are also reported.

For large system configurations where a large number of disks are configured, the system can be configured to avoid collecting physical disk input/output statistics when the **iostat** command is not executing. If the system is configured in the above manner, the first Disk report displays the message `Disk History Since Boot Not Available` instead of the disk statistics. Subsequent interval reports generated by the **iostat** command contain disk statistics collected during the report interval. Any tty and CPU statistics after boot are unaffected. If a system management command is used to re-enable disk statistics keeping, the first **iostat** command report displays activity from the interval starting at the point that disk input/output statistics were enabled.

System Throughput Report: This report is generated if the **-s** flag is specified. This report provides statistics for the entire system. This report has the following format:

`Kbps`	Indicates the amount of data transferred (read or written) in the entire system in KB per second.
`tps`	Indicates the number of transfers per second issued to the entire system.
`Kb_read`	The total number of KB read from the entire system.
`Kb_wrtn`	The total number of KB written to the entire system.

Adapter Throughput Report: This report is generated if the **-a** flag is specified. This report provides statistics on an adapter-by-adapter basis.This report has the following format:

`Kbps`	Indicates the amount of data transferred (read or written) in the adapter in KB per second.
`tps`	Indicates the number of transfers per second issued to the adapter.
`Kb_read`	The total number of KB read from the adapter.
`Kb_wrtn`	The total number of KB written to the adapter.

Disk Input/Output History: To improve performance on, the collection of disk input/output statistics has been disabled. To enable the collection of this data, type:

```
chdev -l sys0 -a iostat=true
```

To display the current settings, type:

```
lsattr -E -l sys0 -a iostat
```

If the collection of disk input/output history is disabled, the first disk report of iostat output displays the message `Disk History Since Boot Not Available` instead of disk statistics. As before, subsequent interval reports generated by the **iostat** command contain disk statistics collected during the report interval.

Flags

-a Displays the adapter throughput report.

-d The **-d** flag is exclusive of the **-t** flag and displays only the disk utilization report.

-s Displays the system throughput report.

-m The **-m** flag will print the path statistics for the following

- Paths to MPIO (Multi-Path I/O) enabled devices.
- Paths in the ESS machines.

The throughput is per device. The throughout for all the paths to that device follow the throughput of that device.

For ESS machines, the vpaths will be treated as disks and hdisks will be treated as Paths. Internally the vpaths are actually disks and hdisks are the paths to them. For MPIO enabled devices, the path name will be represented as **Path0**, **Path1**, **Path2** and so on. The numbers 0, 1, 2, and so on are the path IDs provided by the **lspath** command. Since paths to a device can be attached to any adapter, the adapter report will report the path statistics under each adapter. The disk name will be a prefix to all the paths. For all MPIO enabled devices, the adapter report will print the path names as **hdisk10_Path0**, **hdisk0_Path1**, and so on. For all ESS Machines, the adapter report will print the path names as **vpath0_hdisk3**, **vpath10_hdisk25**, and so on.

-t The **-t** flag is exclusive of the **-d** flag and displays only the tty and cpu usage reports.

-T Prints the time-stamp next to each line of output of **iostat**. The time-stamp displays in the *HH:MM:SS* format.

Notes:

- The **-s** and **-a** flags can both be specified to display both the system and adapter throughput reports.
- If the **-a** flag is specified with the **-t** flag, the tty and CPU report is displayed, followed by the adapter throughput report. Disk Utilization reports of the disks connected to the adapters, will not be displayed after the Adapter throughput report.
- If the **-a** flag is specified with the **-d** flag, tty and CPU report will not be displayed. If Physical Volume parameter is specified, the Disk Utilization Report of the specified Physical volume will be printed under the corresponding adapter to which it belongs.

Examples

1. To display a single history since boot report for all tty, CPU, and Disks, type:

   ```
   iostat
   ```

2. To display a continuous disk report at two second intervals for the disk with the logical name `disk1`, type:

   ```
   iostat -d disk1 2
   ```

3. To display six reports at two second intervals for the disk with the logical name `disk1`, type:

   ```
   iostat disk1 2 6
   ```

4. To display six reports at two second intervals for all disks, type:

   ```
   iostat -d 2 6
   ```

5. To display six reports at two second intervals for three disks named `disk1`, `disk2`, `disk3`, type:

   ```
   iostat disk1 disk2 disk3 2 6
   ```

6. To print the System throughput report, type:

   ```
   iostat -s
   ```

7. To print the Adapter throughput report, type:

   ```
   iostat -a
   ```

8. To print the System and Adapter throughput reports, with only the tty and CPU report (no disk reports), type:

```
iostat -sat
```

9. To print the System and Adapter throughput reports with the Disk Utilization reports of hdisk0 and hdisk7, enter

```
iostat -sad hdisk0 hdisk7
```

10. To display time stamp next to each line of output of **iostat**, type:

```
iostat -T
```

File

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

Related Information

The **vmstat** command.

The **/dev/kmem** special file.

The **knlist** subroutine, **sysconfig** subroutine.

Monitoring and Tuning Disk I/O in *AIX 5L Version 5.2 Performance Management Guide*

The Input and Output Handling Programmer's Overview in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs* describes the files, commands, and subroutines used for low-level, stream, terminal, and asynchronous I/O interfaces.

---

ipcrm Command

Purpose

Removes message queue, semaphore set, or shared memory identifiers.

Syntax

ipcrm [**-m** *SharedMemoryID*] [**-M** *SharedMemoryKey*] [**-q** *MessageID*] [**-Q** *MessageKey*] [**-s** *SemaphoreID*] [**-S** *SemaphoreKey*]

Description

The **ipcrm** command removes one or more message queues, semaphore sets, or shared memory identifiers.

Flags

-m *SharedMemory ID*	Removes the shared memory identifier *SharedMemoryID*. The shared memory segment and data structure associated with *SharedMemoryID* are also removed after the last detach operation.
-M *SharedMemoryKey*	Removes the shared memory identifier, created with the key *SharedMemoryKey*. The shared memory segment and data structure associated with it are also removed after the last detach operation.
-q *MessageID*	Removes the message queue identifier *MessageID* and the message queue and data structure associated with it.
-Q *MessageKey*	Removes the message queue identifier, created with the key *MessageKey*, and the message queue and data structure associated with it.
-s *SemaphoreID*	Removes the semaphore identifier *SemaphoreID* and the set of semaphores and data structure associated with it.

-S *SemaphoreKey* Removes the semaphore identifier, created with the key *SemaphoreKey*, and the set of semaphores and data structure associated with it.

The **msgctl**, **shmctl**, and **semctl** subroutines provide details of the remove operations. The identifiers and keys can be found by using the **ipcs** command.

Examples

To remove the shared memory segment associated with *SharedMemoryID* 18602, enter:

```
ipcrm -m 18602
```

Related Information

The **ipcs** command.

The **msgget** subroutine, **semctl** subroutine, **semget** subroutine, **shmctl** subroutine, **shmget** subroutine.

The Commands Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

ipcs Command

Purpose

Reports interprocess communication facility status.

Syntax

ipcs [**-m**] [**-q**] [**-s**] [**-S**] [**-P**] [**-l**] [**-a** | **-b -c -o -p -t**] [**-T**] [**-C** *CoreFile*] [**-N** *Kernel*]

Description

The **ipcs** command writes to the standard output information about active interprocess communication facilities. If you do not specify any flags, the **ipcs** command writes information in a short form about currently active message queues, shared memory segments, semaphores, remote queues, and local queue headers.

The column headings and the meaning of the columns in an **ipcs** command listing follow. The letters in parentheses indicate the flags that cause the corresponding heading to appear. The designator **all** means the heading is always displayed. These flags only determine what information is provided for each facility. They do not determine which facilities are listed.

T (**all**) the type of facility. There are three facility types:

- **q** message queue
- **m** shared memory segment
- **s** semaphore

ID (**all**) the identifier for the facility entry.

KEY (**all**) the key used as a parameter to the **msgget** subroutine, the **semget** subroutine, or the **shmget** subroutine to make the facility entry.

> **Note:** The key of a shared memory segment is changed to **IPC_PRIVATE** when the segment is removed until all processes attached to the segment detach from it.

MODE (**all**) the facility access modes and flags. The mode consists of 11 characters that are interpreted as follows:

The first two characters can be the following:

R If a process is waiting on a **msgrcv** system call.

S If a process is waiting on a **msgsnd** system call.

D If the associated shared memory segment has been removed. It disappears when the last process attached to the segment detaches it.

C If the associated shared memory segment is to be cleared when the first attach is run.

- If the corresponding special flag is not set.

The next nine characters are interpreted as three sets of 3 bits each. The first set refers to the owner's permissions; the next to permissions of others in the user group of the facility entry; and the last to all others. Within each set, the first character indicates permission to read, the second character indicates permission to write or alter the facility entry, and the last character is currently unused.

The permissions are indicated as follows:

r If read permission is granted.

w If write permission is granted.

a If alter permission is granted.

- If the indicated permission is *not* granted.

OWNER (**all**) The login name of the owner of the facility entry.

GROUP (**all**) The name of the group that owns the facility entry.

CREATOR (**a**,**c**) The login name of the creator of the facility entry.

CGROUP (**a**,**c**) The group name of the creator of the facility entry.

Note: For the **OWNER**, **GROUP**, **CREATOR**, and **CGROUP**, the user and group IDs display instead of the login names.

CBYTES (**a**,**o**) The number of bytes in messages currently outstanding on the associated message queue.

QNUM (**a**,**o**) The number of messages currently outstanding on the associated message queue.

QBYTES (**a**,**b**) The maximum number of bytes allowed in messages outstanding on the associated message queue.

LSPID (**a**,**p**) The ID of the last process that sent a message to the associated queue. If the last message sent was from a process in a node other than the node that holds the queue, **LSPID** is the PID of the kernel process that actually placed the message on the queue, not the PID of the sending process.

LRPID (**a**,**p**) The ID of the last process that received a message from the associated queue. If the last message received was from a process in a node other than the node that holds the queue, **LRPID** is the PID of the kernel process that actually received the message on the queue, not the PID of the receiving process.

STIME (**a**,**t**) The time when the last message was sent to the associated queue. For remote queues, this is the server time. No attempt is made to compensate for time-zone differences between the local clock and the server clock.

RTIME (**a**,**t**) The time when the last message was received from the associated queue. For remote queues, this is the server time. No attempt is made to compensate for any time-zone differences between the local clock and the server clock.

CTIME (**a**,**t**) The time when the associated entry was created or changed. For remote queues, this is the server time. No attempt is made to compensate for any time-zone differences between the local clock and the server clock.

NATTCH (**a**,**o**) The number of processes attached to the associated shared memory segment.

SEGSZ (**a**,**b**) The size of the associated shared memory segment.

CPID (**a**,**p**) The process ID of the creator of the shared memory entry.

LPID	(**a**,**p**) The process ID of the last process to attach or detach the shared memory segment.
ATIME	(**a**,**t**) The time when the last attach was completed to the associated shared memory segment.
DTIME	(**a**,**t**) The time the last detach was completed on the associated shared memory segment.
NSEMS	(**a**,**b**) The number of semaphores in the set associated with the semaphore entry.
OTIME	(**a**,**t**) The time the last semaphore operation was completed on the set associated with the semaphore entry.
SID	(**S**) The shared memory segment id. **SID**s can be used as input to the **svmon -S** command.

This command supports multibyte character sets.

Flags

-a	Uses the **-b**, **-c**, **-o**, **-p** and **-t** flags.
-b	Writes the maximum number of bytes in messages on queue for message queues, the size of segments for shared memory, and the number of semaphores in each semaphores set.
-c	Writes the login name and group name of the user that made the facility.
-C*CoreFile*	Uses the file specified by the *CoreFile* parameter in place of the **/dev/mem** file. The *CoreFile* parameter is a memory image file produced by the Ctrl-(left)Alt-Pad1 key sequence.
-l	When used with the **-S** flag, this flag writes the list of **SID**s unwrapped.
-m	Writes information about active shared memory segments.
-N*Kernel*	Uses the specified *Kernel* (the **/usr/lib/boot/unix** file is the default).
-o	Writes the following usage information: • Number of messages on queue • Total number of bytes in messages in queue for message queues • Number of processes attached to shared memory segments
-p	Writes process number information: • Process number of the last process to receive a message on message queues • Process number of last process to send a message on message queues • Process number of the creating process • Process number of last process to attach or detach on shared memory segments
-P	Writes the list of SIDs (segment IDs) associated with the shared memory ID, along with the number of bytes pinned to that segment and an indication of whether the segment is large-page enabled or not. If the segment is large-page enabled, a 'Y' is displayed, otherwise a '-' is displayed.
-q	Writes information about active message queues.
-s	Writes information about active semaphore set.
-S	Writes the list of **SID** attached to shared memory id.
-t	Writes time information: • Time of the last control operation that changed the access permissions for all facilities • Time of the last **msgsnd** and **msgrcv** on message queues • Time of the last **shmat** and **shmdt** on shared memory • Time of the last **semop** on semaphore sets
-T	Writes the output of the **-t** flag with the date.

Notes:

1. If the user specifies either the **-C** or **-N** flag, the real and effective UID/GID is set to the real UID/GID of the user invoking **ipcs**.
2. Values can change while **ipcs** is running; the information it gives is guaranteed to be accurate only when it was retrieved.

Example

Example output from entering `ipcs` without flags:

```
IPC status from /dev/mem as of Mon Aug 14 15:03:46 1989
T    ID         KEY        MODE       OWNER     GROUP
Message Queues:
q       0    0x00010381 -Rrw-rw-rw-   root      system
q   65537    0x00010307 -Rrw-rw-rw-   root      system
q   65538    0x00010311 -Rrw-rw-rw-   root      system
q   65539    0x0001032f -Rrw-rw-rw-   root      system
q   65540    0x0001031b -Rrw-rw-rw-   root      system
q   65541    0x00010339--rw-rw-rw-    root      system
q       6    0x0002fe03 -Rrw-rw-rw-   root      system
Shared Memory:
m   65537    0x00000000 DCrw-------   root      system
m  720898    0x00010300 -Crw-rw-rw-   root      system
m   65539    0x00000000 DCrw-------   root      system
Semaphores:
s  131072    0x4d02086a --ra-ra----   root      system
s   65537    0x00000000 --ra-------   root      system
s 1310722    0x000133d0 --ra-------   7003      30720
```

Files

/usr/lib/boot/unix	Specifies the system kernel image.
/dev/mem	Specifies memory.
/etc/passwd	Specifies user names.
/etc/group	Specifies group names.
/usr/include/sys/ipc.h	Contains the header file.

Related Information

The **ipcrm** command.

The **svmon** command.

The **msgrcv** subroutine, **msgsnd** subroutine, **semop** subroutine, **shmat** subroutine, **shmdt** subroutine.

Commands Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

ipfilter Command

Purpose

Extracts different operation headers from an ipreport output file and displays them in a table. Some customized nfs information regarding requests and replies is also provided.

Syntax

ipfilter [-f [u n t x c a]] [-s [u n t x c a]] [-n [-d *milliseconds*]] *ipreport_output_file*

Description

The **ipfilter** command extracts specific information from an ipreport output file and displays it to a table. The operation headers currently recognized are: udp, nfs, tcp, ipx, icmp, atm. The ipfilter command has three different types of reports:

- A single file (**ipfilter.all**) that displays a list of all the selected operations. The table displays packet number, Time, Source and Destination, Length, Sequence #, Ack #, Source Port, Destination Port, Network Interface, and Operation Type.

- Individual files for each selected header (**ipfilter.udp**, **ipfilter.nfs**, **ipfilter.tcp**, **ipfilter.ipx**, **ipfilter.icmp**, **ipfilter.atm**). The information is the same as **ipfilter.all**.
- A file **nfs.rpt** that reports on nfs requests and replies. The table contains: Transaction ID #, Type of Request, Status of Request, Call Packet Number, Time of Call, Size of Call, Reply Packet Number, Time of Reply, Size of Reply, and Elapsed millisecond between call and reply.

Flags

u n t x c a	Specifies operation headers (udp, nfs, tcp, ipx, and icmp and atm respectively).
-d *milliseconds*	Only Call/Reply pairs whose elapsed time is greater than *milliseconds* are to be shown.
-f [u n t x c a]	Selected operations are to be shown in **ipfilter.all**.
-n	Generates an **nfs.rpt**.
-s [u n t x c]	Separate files are to be produced for each of the selected operations.

Related Information

The **iptrace** daemon, **ipreport** command.

---

ipreport Command

Purpose

Generates a packet trace report from the specified packet trace file.

Syntax

/usr/sbin/ipreport [**-e**] [**-r**] [**-n**] [**-s**] *LogFile*

Description

The **/usr/sbin/ipreport** command generates a trace report from the specified trace file created by the **iptrace** command. The *LogFile* parameter specifies the name of the file containing the results of the Internet Protocol trace. This file is created by the **iptrace** command.

Flags

-e Generates the trace report in EBCDIC format. The default format is ASCII.
-r Decodes remote procedure call (RPC) packets.
-n Includes a packet number to facilitate easy comparison of different output formats.
-s Prepends the protocol specification to every line in a packet.

Related Information

The **iptrace** command, **trpt** command.

---

ipsec_convert Command

Purpose

Converts IP Security tunnel export files to a format that can be imported by the IBM Secure Network Gateway.

Syntax

ipsec_convert SNG22 | FW31 [**-f** *export_directory*]

Description

IP Security allows the importing of IBM Secure Network Gateway 2.2 and IBM Firewall 3.1 tunnels using the **imptun** command. However, these firewall products do not allow the reverse capability. The **ipsec_convert** command allows for this capability by translating exported IP Security tunnels to IBM Firewall tunnels. The translated files will be placed in the current directory.

Flags

SNG22 | FW31	Specifies whether the format of the resulting files will be in the format of IBM Secure Network Gateway 2.2 or IBM Firewall 3.1 format.
-f	Specifies the directory where the exported IPSec files are located.

Related Information

The **imptun** command.

---

ipsecstat Command

Purpose

Lists status of IP Security devices, IP Security crypto algorithms, and statistics of IP Security packets.

Syntax

ipsecstat [**-c**] [**-d**] [**-A**] [**-E**]

Description

The **ipsecstat** command, used without flags, displays the status of the IP Security devices, the crypto algorithms installed for IP Security, and the statistics of IP Security packets.

The command can be used with flags to only list the status of IP Security devices, to only list the installed algorithms, or to reset statistic counters (to zero).

Flags

-c	Resets statistics counters (after displaying current value). The **-c** flag cannot be used with any other flags.
-d	Lists only the status of the IP Security devices. The **-d** flag cannot be used with any other flags.
-A	Lists only the installed authentication algorithms. The **-A** flag cannot be used with any other flags.
-E	Lists only the installed encryption algorithms. The **-E** flag cannot be used with any other flags.

---

ipsectrcbuf Command

Purpose

Lists the contents of tracing buffers in the IP Security subsystem.

Syntax

ipsectrcbuf [**-l** {**0**|**1**|**2**}]

Description

The IP Security subsystem maintains a memory resident trace buffer to help debug if there is a problem. The content of the buffer, a fixed number of the most recent trace messages, will be in a system dump or can be listed by running this command with no arguments.

Flags

-l

Sets the IP Security trace level. By default, of the nine IP Security trace hooks, only IPSEC_ERROR trace messages are put into the buffer. To enable or disable the other trace hooks, use the **-l** flag with one of the following values:

0 Only IPSEC_ERROR trace messages are written to the buffer. This is the default.

1 IPSEC_FILTER, IPSEC_CAPSUL, IPSEC_CRYPTO, IPSEC_TUNNEL, as well as IPSEC_ERROR trace messages are written to the buffer.

2 All IP Security trace messages are put into the buffer (that includes IPSEC_FILTER_INFO, IPSEC_CAPSUL_INFO, IPSEC_CRYPTO_INFO, and IPSEC_TUNNEL_INFO as well as those in level 1).

---

iptrace Daemon

Purpose

Provides interface-level packet tracing for Internet protocols.

Syntax

/usr/sbin/iptrace [**-a**] [**-b**][**-e**] [**-u**] [**-P***Protocol_list*] [**-i***Interface*] [**-p***Port_list*] [**-s***Host* [**-b**]] [**-d***Host*] [**-L** *Log_size*] *LogFile*

Description

The **/usr/sbin/iptrace** daemon records Internet packets received from configured interfaces. Command flags provide a filter so that the daemon traces only packets meeting specific criteria. Packets are traced only between the local host on which the **iptrace** daemon is invoked and the remote host.

If the **iptrace** process was started from a command line without the System Resource Controller (SRC), it must be stopped with the **kill -15** command. The kernel extension loaded by the **iptrace** daemon remains active in memory if **iptrace** is stopped any other way.

The *LogFile* parameter specifies the name of a file to which the results of the **iptrace** command are sent. To format this file, run the **ipreport** command. The **ipreport** command may display the message `TRACING DROPPED xxxx PACKETS`. This count of dropped packets indicates only the number of packets that the **iptrace** command was unable to grab because of a large packet, the size of which exceeded the socket-receive buffer size. This message does NOT mean that the packets are being dropped by the system.

Note: The file specified by the *LogFile* parameter should not reside on an NFS-mounted file system. Specifying an output file on an NFS-mounted file system can cause the **iptrace** daemon to hang. You may be unable to kill the **iptrace** daemon if it hangs, requiring that you restart the system.

Flags

-a	Suppresses ARP packets.
-b	Changes the **-d** or **-s** flags to bidirectional mode.
-d *Host*	Records packets headed for the destination host specified by the *Host* variable. The *Host* variable can be a host name or an Internet address in dotted-decimal format. If used with the **-b** flag, the **-d** flag records packets both going to and coming from the host specified by the *Host* variable.
-e	Enables promiscuous mode on network adapters that support this function.
-i *Interface*	Records packets received on the interface specified by the *Interface* variable.
-L Log_size	This option causes **iptrace** to log data in such that the `LogFile` is copied to `LogFile.old` at the start and also every time it becomes approximately *Log_size* bytes long.
-P *Protocol_list*	Records packets that use the protocol specified by the *Protocol_list* variable which is a comma separated list of protocols. The Protocols can be a decimal number or name from the **/etc/protocols** file.
-p *Port_list*	Records packets that use the port number specified by the *Port_list* variable which is a comma separated list of ports. The *Port_list* variable can be a decimal number or name from the **/etc/services** file.
-s *Host*	Records packets coming from the source host specified by the *Host* variable. The *Host* variable can be a host name or an Internet address in dotted-decimal format. If used with the **-b** flag, the **-s** flag records packets both going to and coming from the host specified by the *Host* variable.
-u	Unloads the kernel extension that was loaded by the **iptrace** daemon at startup.

Examples

1. To start the iptrace daemon with the System Resource Controller (SRC), enter:

   ```
   startsrc -s iptrace -a "/tmp/nettrace"
   ```

   To stop the iptrace daemon with SRC enter the following:

   ```
   stopsrc -s iptrace
   ```

2. To record packets coming in and going out to any host on every interface, enter the command in the following format:

   ```
   iptrace /tmp/nettrace
   ```

   The recorded packets are received on and sent from the local host. All packet flow between the local host and all other hosts on any interface is recorded. The trace information is placed into the `/tmp/nettrace` file.

3. To record packets received on an interface from a specific remote host, enter the command in the following format:

   ```
   iptrace - i en0 -p telnet -s airmail /tmp/telnet.trace
   ```

   The packets to be recorded are received on the `en0` interface, from remote host`airmail`, over the `telnet` port. The trace information is placed into the `/tmp/telnet.trace` file.

4. To record packets coming in and going out from a specific remote host, enter the command in the following format:

   ```
   iptrace -i en0 -s airmail -b /tmp/telnet.trace
   ```

The packets to be recorded are received on the `en0` interface, from remote host`airmail`. The trace information is placed into the `/tmp/telnet.trace` file.

Related Information

The **ipreport** command, the **tcpdump** command.

The **/etc/protocols** file format, **/etc/services** file format.

---

isC2host Command

Purpose

Determine the C2 status of a system.

Syntax

isC2host [**-i** | **-s**]

Description

The **isC2host** command returns the configuration status of the host machine. If the host has been configured to operate in C2 mode, the command exits with a zero (true) code. If the host has not been configured to operate in C2 mode, the command exits with a non-zero (false) code.

This command may be used in shell scripts where the security status of the host must be known.

The **-i** option is used to determine the installation status of the system. The C2 status of the system is determined by examining the ODM database, and the exit status indicates whether or not the system was installed in C2 mode.

The **-s** option is used to initialize AIX in C2 mode and may only be issued by the root user. The C2 status of the system is determined by examining the ODM database. On a system that has not been installed with C2, as indicated by the ODM, this option performs no operation.

Flags

-i	Determine the C2 installation status of the system.
-s	Set the C2 status of the system from the ODM.

Subcommands

Exit Status

0 When used with no options, the system has been initialized to operate in C2 mode. When used with the **-s** flag, the system was successfully initialized according to the C2 mode setting defined in the ODM database. When used with the **-i** flag, the system was installed with C2 enabled.

1 When used with no options, the system has not been initialized to operate in C2 mode. When used with the **-s** flag, the system could not be initialized to operate in the security mode that was defined in the ODM. When used with the **-i** flag, the system was installed with C2 enabled but is not currently operating in C2 mode.

2 When used with the **-s** option, the **isC2host** command was executed by a non-root user. When used with the **-i** option, the system was not installed with C2 enabled.

3 The **isC2host** command was executed with an invalid command line option.

Files

/usr/sbin/isC2host	Contains the **isC2host** command.

Related Information

The **chC2admin** command, **lsC2admin** command, **mkC2admin** command, **rmC2admin** command.

The **sysconfig()** subroutine.

The **Object Data Manager** subsystem.

---

isCChost Command

Purpose

Determine the Common Criteria enabled status of a system.

Syntax

isCChost [**-i** | **-s**]

Description

The **isCChost** command returns the configuration status of the host machine. If the host has been configured to operate in Common Criteria enabled mode, the command exits with a zero (true) code. If the host has not been configured to operate in Common Criteria enabled mode, the command exits with a non-zero (false) code.

This command may be used in shell scripts where the security status of the host must be known.

The **-i** option is used to determine the installation status of the system. The Common Criteria enabled status of the system is determined by examining the ODM database, and the exit status indicates whether or not the system was installed in Common Criteria enabled mode.

The **-s** option is used to initialize AIX in Common Criteria enabled mode and may only be issued by the root user. The Common Criteria enabled status of the system is determined by examining the ODM database. On a system that has not been installed with Common Criteria enabled, as indicated by the ODM, this option performs no operation.

Flags

-i	Determine the Common Criteria enabled installation status of the system.
-s	Set the Common Criteria enabled status of the system from the ODM.

Subcommands

Exit Status

0 When used with no options, the system has been initialized to operate in Common Criteria enabled mode. When used with the **-s** flag, the system was successfully initialized according to the Common Criteria enabled mode setting defined in the ODM database. When used with the **-i** flag, the system was installed with Common Criteria enabled enabled.

1 When used with no options, the system has not been initialized to operate in Common Criteria enabled mode. When used with the **-s** flag, the system could not be initialized to operate in the security mode that was defined in the ODM. When used with the **-i** flag, the system was installed with Common Criteria enabled but is not currently operating in Common Criteria enabled mode.

2 When used with the **-s** option, the **isCChost** command was executed by a non-root user. When used with the **-i** option, the system was not installed with Common Criteria enabled.

3 The **isCChost** command was executed with an invalid command line option.

Files

/usr/sbin/isCChost	Contains the **isCChost** command.

Related Information

The **chCCadmin** command, **lsCCadmin** command, **mkCCadmin** command, **rmCCadmin** command.

The **sysconfig()** subroutine.

The **Object Data Manager** subsystem.

---

istat Command

Purpose

Examines i-nodes.

Syntax

istat {*FileName* | *i-nodeNumber Device*}

Description

The **istat** command displays the i-node information for a particular file. You can specify the file either by providing a file or directory name with the *FileName* parameter or by providing an i-node number with the *i-nodeNumber* parameter and a device name with the *Device* parameter. You can specify the *Device* parameter as either a device name or as a mounted file system name.

If you specify the *FileName* parameter, the **istat** command writes the following information about the file:

- Device where the file resides
- i-node number of the file, on that device
- File type, such as normal, directory, and block device
- File access permissions
- Name and identification number of the owner and group

  Note: The owner and group names for remote files are taken from the local **/etc/passwd** file.
- Number of links to the file
- If the i-node is for a normal file, length of the file
- If the i-node is for a device, major and minor device designations
- Date of the last i-node update
- Date of the last file modification
- Date of the last reference to the file

If you specify the *i-nodeNumber* and *Device* parameters, the **istat** command also displays, in hexadecimal values, the block numbers recorded in the i-node.

Note: The *Device* parameter cannot refer to a remote device.

Examples

1. To display the information in the i-node corresponding to the `/usr/bin/ksh` file, enter:

   ```
   istat /usr/bin/ksh
   ```

   This command displays the i-node information for the `/usr/bin/ksh` file. The information looks similar to the following:

   ```
   Inode 10360 on device 10/6    File
   Protection: r-xr-xr-x
   Owner: 2(bin)     Group: 2(bin)
   Link count: 2     Length 372298 bytes

   Last updated:  Wed May 13 14:08:13 1992
   Last modified: Wed May 13 13:57:00 1992
   Last accessed: Sun Jan 31 15:49:23 1993
   ```

2. To display i-node information by specifying a file i-node number, enter:

   ```
   istat 10360 /dev/hd2
   ```

   This command displays the information contained in the i-node identified by the number `10360` on the `/dev/hd2` device. In addition to the information shown in Example 1, this displays:

   ```
   Block pointers (hexadecimal):
   2a9a   2a9b   2a9c   2a9d   2a9e   2a9f   2aa0   2aa1
   ```

   These numbers are addresses of the disk blocks that make up the **/usr/bin/ksh** file.

Files

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

Related Information

The **fsdb** command.

The **filesystems** file, **jfs/filsys.h** file.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* provides information on working with files.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* provides an introduction on i-nodes and how they are used by the file system.

jobs Command

Purpose

Displays status of jobs in the current session.

Syntax

jobs [**-l** | **-n** | **-p**] [*JobID* ...]

Description

The **jobs** command displays the status of jobs started in the current shell environment. If no specific job is specified with the *JobID* parameter, status information for all active jobs is displayed. If a job termination is reported, the shell removes that job's process ID from the list of those known by the current shell environment.

The **/usr/bin/jobs** command does not work when operating in its own command execution environment, because that environment does not have applicable jobs to manipulate. For this reason, the **jobs** command is implemented as a Korn shell or POSIX shell regular built-in command.

If the **-p** flag is specified, output consists of one line for each process ID. If no flags are specified, standard output is a series of lines with the following fields:

`job-number` Indicates the process group number to use with the **wait**, **fg**, **bg**, and **kill** commands. When used with these commands, prefix the job number with a % (percent sign).

`current` A + (plus sign) identifies the job that will be used as a default for the **fg** or **bg** commands. This job ID can also be specified using the **%+** (percent sign, plus) or **%%** (double percent sign).

A - (minus sign) identifies the job that becomes the default if the current default job exits. This job ID can also be specified using **%-** (percent sign, minus).

For other jobs, the `current` field is a space character. Only one job can be identified with a +, and only one job can be identified with a -. If there is a single suspended job, that will be the current job. If there are at least two suspended jobs, then the previous job is also suspended.

`state` Displays one of the following values (in the POSIX locale):

Running
Indicates that the job has not been suspended by a signal and has not exited.

Done Indicates that the job completed and returned exit status 0.

Done (*code***)**
Indicates that the job completed normally and that it exited with the specified non-zero exit status code. This code is expressed as a decimal number.

Stopped
Indicates that the job was suspended.

Stopped (SIGTSTP)
Indicates that the **SIGTSTP** signal suspended the job.

Stopped (SIGSTOP)
Indicates that the **SIGSTOP** signal suspended the job.

Stopped (SIGTTIN)
Indicates that the **SIGTTIN** signal suspended the job.

Stopped (SIGTTOU)
Indicates that the **SIGTTOU** signal suspended the job.

`command` The associated command that was given to the shell.

If the **-l** flag is specified, a field containing the process group ID is inserted before the `state` field. Also, more processes in a process group may be output on separate lines, using only the `job-number` and `command` fields.

Flags

-l	(lowercase L) Provides more information about each job listed. This information includes the job number, current job, process group ID, state, and the command that initiated the job.
-n	Displays only jobs that have stopped or exited since last notified.
-p	Displays the process IDs for the process group leaders for the selected jobs.

By default the **jobs** command displays the status of all stopped jobs, all running background jobs, and all jobs whose status has changed but not been reported by the shell.

Exit Status

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

Examples

1. To display the status of jobs in the current environment, enter:

   ```
   jobs -l
   ```

   The screen displays a report similar to the following output:

   ```
   +[4] 139  Running        CC - C foo c&
   -[3] 465  Stopped        mail morris
    [2] 687  Done(1)        foo.bar&
   ```

2. To display the process ID for the job whose name begins with ″m,″ enter:

   ```
   job -p %m
   ```

   Using the jobs reported in Example 1, the screen displays the following process ID:

   ```
   465
   ```

Files

/usr/bin/ksh	Contains the Korn shell **jobs** built-in command.
/usr/bin/jobs	Contains the **jobs** command.

Related Information

The **bg** command,**csh** command, **fg** command, **kill** command, **ksh** command, **wait** command.

---

join Command

Purpose

Joins the data fields of two files.

Syntax

join [**-a** *FileNumber* | **-v** *FileNumber*] [**-e** *String*] [**-o** *List*] [**-t** *Character*] [**-1** *Field*] [**-2** *Field*] *File1 File2*

Description

The **join** command reads the files specified by the *File1* and *File2* parameters, joins lines in the files according to the flags, and writes the results to standard output. The *File1* and *File2* parameters must be text files. Both *File1* and *File2* must be sorted in the collating sequence of sort -b on the field that they are being joined by before invoking the **join** command.

One line appears in the output for each identical join field appearing in both files. The join field is the field in the input files examined by the **join** command to determine what will be included in the output. The output line consists of the join field, the rest of the line from the file specified by the *File1* parameter, and the rest of the line from the file specified by the *File2* parameter. Specify standard input in place of either the *File1* or *File2* parameter by substituting a **-** (dash) as the file name. Both input files cannot be specified with a **-** (dash).

Fields are usually separated by a space, a tab character, or a new-line character. In this case, the **join** command treats consecutive separators as one and discards leading separators.

Flags

-1 *Field*
Joins the two files using the field specified by the *Field* variable in the *File1* input file. The value of the *Field* variable must be a positive decimal integer.

-2 *Field*
Joins the two files using the field specified by the *Field* variable in the *File2* input file. The value of the *Field* variable must be a positive decimal integer.

-a *FileNumber*
Produces an output line for each line in the file specified by the *FileNumber* variable whose join fields do not match any line in the other input file. The output lines are produced in addition to the default output. The value of the *FileNumber* variable must be either 1 or 2, corresponding to the files specified by the *File1* and *File2* parameters, respectively. If this flag is specified with the **-v** flag, this flag is ignored.

-e *String*
Replaces empty output fields with the string specified by the *String* variable.

-o *List*
Constructs an output line to comprise the fields specified in the *List* variable. One of the following forms applies to the *List* variable:

FileNumber.Field
Where *FileNumber* is a file number and *Field* is a decimal-integer field number. Separate multiple fields with a **,** (comma) or space characters with quotation marks around the multiple fields.

0 **(zero)**
Represents the join field. The **-o** *0* flag essentially selects the union of the join fields.

-t *Character*
Uses the character specified by the *Character* parameter as the field separator character in the input and the output. Every appearance of the character in a line is significant. The default separator is a space. With default field separation, the collating sequence is that of the **sort -b** command. If you specify **-t**, the sequence is that of a plain sort. To specify a tab character, enclose it in single quotation marks.

-v *FileNumber*
Produces an output line for each line in the file specified by the *FileNumber* variable whose join fields do not match any line in the other input file. Default output is not produced. The value of the *FileNumber* variable must be either 1 or 2, corresponding to the files specified by *File1* and *File2* parameters, respectively. If this flag is specified with the **-a** flag, the **-a** flag is ignored.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Examples

Note: The vertical alignment shown in the following examples might not be consistent with your output.

1. To perform a simple join operation on two files where the first fields are the same, type:

   ```
   join phonedir names
   ```

   If the `phonedir` file contains the following names:

   ```
   Adams A.        555-6235
   Dickerson B.    555-1842
   Erwin G.        555-1234
   Jackson J.      555-0256
   Lewis B.        555-3237
   Norwood M.      555-5341
   Smartt D.       555-1540
   Wright M.       555-1234
   Xandy G.        555-5015
   ```

   and the `names` file contains these names and department numbers:

   ```
   Erwin           Dept. 389
   Frost           Dept. 217
   Nicholson       Dept. 311
   Norwood         Dept. 454
   Wright          Dept. 520
   Xandy           Dept. 999
   ```

   the **join** command displays:

   ```
   Erwin G.        555-1234        Dept. 389
   Norwood M.      555-5341        Dept. 454
   Wright M.       555-1234        Dept. 520
   Xandy G.        555-5015        Dept. 999
   ```

   Each line consists of the join field (the last name), followed by the rest of the line found in the `phonedir` file and the rest of the line in the `names` file.

2. To display unmatched lines with the **join** command, type:

   ```
   join  -a1 phonedir names
   ```

   If the `phonedir` and `names` files are the same as in Example 1, the **join** command displays:

   ```
   Erwin G.        555-1234         Dept. 389
   Frost                            Dept. 217
   Nicholson                        Dept. 311
   Norwood M.      555-5341         Dept. 454
   Wright M.       555-1234         Dept. 520
   Xandy G.        555-5015         Dept. 999
   ```

   This command performs the same join operation as in Example 1, and also lists the lines of names that have no match in the `phonedir` file. The names `Frost` and `Nicholson` are included in the listing, even though they do not have entries in the `phonedir` file.

3. To display selected fields with the **join** command, type:

   ```
   join  -o 2.3,2.1,1.2,1.3 phonedir names
   ```

   This displays the following fields in the order given:

Field 3 of `names`	Department number
Field 1 of `names`	Last name
Field 2 of `phonedir`	First initial

Field 3 of phonedir	Telephone number

If the phonedir file and names files are the same as in Example 1, the **join** command displays:

```
389             Erwin G.        555-1234
454             Norwood M.      555-5341
520             Wright M.       555-1234
999             Xandy G.        555-5015
```

4. To perform the join operation on a field other than the first, type:

   ```
   sort +2 -3 phonedir | join  -1 3 - numbers
   ```

   This command combines the lines in the phonedir and numbers files, comparing the third field of the phonedir file to the first field of the numbers file.

   First, this command sorts the phonedir file by the third field, because both files must be sorted by their join fields. The output of the **sort** command is then piped to the **join** command. The **-** (dash) by itself causes the **join** command to use this output as its first file. The -1 3 flag defines the third field of the sorted phonedir file as the join field. This is compared to the first field of numbers because its join field is not specified with a **-2** flag.

   If the numbers file contains:

   ```
   555-0256
   555-1234
   555-5555
   555-7358
   ```

   then this command displays the names listed in the phonedir file or each telephone number:

   ```
   555-0256        Jackson J.
   555-1234        Erwin G.
   555-1234        Wright M.
   ```

   Note that the **join** command lists all the matches for a given field. In this case, the **join** command lists both Erwin G. and Wright M. as having the telephone number 555-1234. The number 555-5555 is not listed because it does not appear in the phonedir file.

Files

/usr/bin/join	Contains the **join** command.
/usr/lib/nls/loc/*.src	Contains collation information.

Related Information

The **awk** command, **comm** command, **cut** command, **paste** command, **sort** command.

Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Input and Output Redirection Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

National Language Support Overview for Programming in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

joinvg Command

Purpose
Joins a snapshot volume group back into its orginal volume group.

Syntax
joinvg [**-f**] *VGname*

Description
Joins a snapshot volume group that was created with the **splitvg** command back into its orginal volume group. The snspshot volume group is deleted and the disks reactivated in the orginal volume group. Any stale partitions will be resycrhonized by a background process.

Flags

-f	Forces the join when disks in the snashot volume group are not active. The mirror copy on the inactive disks will be removed from the orginal volume group.
VGname	The orginal volume group name with the **splitvg** command.

Security
Access Control: You must have `root` authority to run this command.

Examples
To join the the original volume group, **testvg**, with the snapshot volume group **snapvg**, type:

```
joinvg testvg
```

Files

/usr/sbin	Directory where the **joinvg** command resides.

Related Information
The **splitvg** and **recreatevg** commands.

kdb Command

Purpose
Displays system images for examining a dump.

Syntax
kdb [*SystemImageFile* [*KernelFile*]]

Description
The **kdb** command is an interactive utility for examining an operating system image or the running kernel. The **kdb** command interprets and formats control structures in the system and provides miscellaneous functions for examining a dump.

The *SystemImageFile* parameter specifies the file that contains the system image. The default *SystemImageFile* is **/dev/pmem**.

The *KernelFile* parameter contains the kernel symbol definitions. The default for the *KernelFile* is **/unix**.

Root permissions are required for execution of the **kdb** command on the active system. This is required because the special file **/dev/pmem** is used. To run the **kdb** command on the active system, type:

```
kdb
```

To invoke the **kdb** command on a system image file, type:

```
kdb SystemImageFile
```

where *SystemImageFile* is either a file name or the name of the dump device. When invoked to view data from a *SystemImageFile* the **kdb** command sets the default thread to the thread running at the time the *SystemImageFile* was created.

Notes:

1. When using the **kdb** command a kernel file must be available.
2. Stack tracing of the current process on a running system does not work

Subcommand Arguments

The following table describes the most common argument types referenced in the subcommand syntax diagrams that follow.

Argument	Description
*	A wildcard used to select all entries.
-	Used to turn page scrolling on or off. It is used with the **setup** subcommand at the **kdb** command prompt.
count	A hex constant specifying the number of times to perform a specific operation.
cpu	A decimal value specifying a cpu number in a SMP machine.
eaddr	Effective address. This may be a hex constant or an expression.
paddr	A physical address.
pid	A hex constant or expression specifying a process ID.
selection	Indicates that a menu is displayed from which a selection must be made.
slot	A decimal constant specifying a slot number within a table.
symb	A symbolic reference to a value. Symbols from the kernel or kernel extensions or both may be used.
tid	A hex constant or expression specifying a thread ID.
tslot	A decimal constant specifying a slot number within the thread table.

Commands in kdb

The following lists the subcommands available within the **kdb** command. Refer to the Subcommands for the KDB Kernel Debugger and kdb Command in the *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts* for more detailed information, including examples, on the subcommands.

kdb allows output redirection of these subcommands via the operators ″|″, ″>″ and ″>>″. The ″|″ symbol pipes all output of the command before the symbol, to the input of the command after the symbol. The ″>″ operator writes the output of the command preceding the operator to the file name following the operator; any existing file is overwritten. The ″>>″ operator appends the output of the command preceding the operator to the file name following the operator.

This output redirection feature is applicable only to the **kdb** command, and not to the KDB Kernel Debugger.

The following subcommand descriptions are grouped into the functional sets:

- Basic subcommands
- Memory display and disassembly subcommands
- Symbol subcommands
- Calculator subcommands
- Machine status subcommands
- Loader subcommands
- Process subcommands
- LVM subcommands
- SCSI subcommands
- Memory allocation subcommands
- File system subcommands
- System table subcommands
- Network subcommands
- VMM subcommands

Basic subcommands

alias [*aliasname* [=*Comand String*]]	This command defines or displays aliases. It also creates or redefines alias definitions or writes existing alias definitions to standard output.
h	Display a list of commands with short descriptions.
his [*?*] [*count*]	The **hist** command prints history commands.
e	Exits from the **kdb** command. This is also recognized by the **e** subcommand alias.
set [*toggle* | *count*]	The **setup** subcommand lists and manipulates kdb toggles.
f [**+x**|**-x**][*tslot* | *eaddr*]	Displays a stack trace for the current or specified thread.
ctx [*cpu*]	The **context** command is used to switch to **kdb** context for a CPU. With no argument, the **kdb** command returns to using the current context.
cdt [*cdt_number* [*cdt_entry*]]	The **cdt** command is used display information about Component Dump Tables (CDTs) or view the data for a component dump entry for a dump.
print [**-l** <*offset*> | <*name*> [**-e** <*end_val*>][<*type*>] <*address*> [**-a** <*count*>] [<*type*>] <*address*>	Helps in interpreting a dump of memory by formatting it into a given C language data structure and displaying it.

Memory display and disassembly subcommands

d *symb*|*eaddr* [*count*]	Display data in byte format.
dw *symb*|*eaddr* [*count*]	Display data in word format.
dd *symb*|*eaddr* [*count*]	Display data in double word format.
dp *paddr* [*count*]	Display data in byte format.
dpw *paddr* [*count*]	Display data in word format.
dpd *paddr* [*count*]	Display data in double word format.
dc *symb*|*eaddr* [*count*]	Display disassembled instruction
dpc *paddr* [*count*]	Display disassembled instruction
dr [*gp*|*sr*|*sp*|*reg_name*]	Display registers. Argument *gp* displays all general purpose registers, *sr* displays all segment register, and *sp* displays all special purpose registers. A specific register may be displayed by name.
di [hexadecimal instr]	Decode the given hexadecimal instruction word with the opcode and the operands.

find [**-s**] *symb*|*eaddr* pattern [*mask* [*delta*]]	Find a pattern. The **-s** option indicates that *pattern* is a string, instead of a hex value.
findp [**-s**] *paddr* pattern [*mask* [*delta*]]	Find a pattern. The **-s** option indicates that *pattern* is a string, instead of a hex value.
ext [**-p**] *eaddr delta* [*size* [*count*]]	The **ext** subcommand displays *size* words starting at *eaddr*, then increments by *delta* words and displays *size* words again. This continues until *count* loops are done. The **-p** flag indicates that *delta* is not the increment value, but the offset to a pointer to the next memory location.
extp [**-p**] *paddr delta* [*size* [*count*]]	The **ext** subcommand displays *size* words starting at *eaddr*, then increments by *delta* words and displays *size* words again. This continues until *count* loops are done. The **-p** flag indicates that *delta* is not the increment value, but the offset to a pointer to the next memory location.

Symbol subcommands

nm *symb*	Translate a symbol to an effective address.
ns	The **ns** toggle may be used to enable/disable symbol translation.
ts *eaddr*	Translate an effective address to a symbol.
which *symbol* | *addr*	Displays the name of the kernel source file containing *symbol* or *addr*.

nm *symb*	Translate a symbol to an effective address.
ns	The **ns** toggle may be used to enable/disable symbol translation.
ts *eaddr*	Translate an effective address to a symbol. Display

Calculator subcommands

hcal *hex_expr*	The **hcal** command may be used to convert hexadecimal values to decimal.
dcal *decimal_expr*	The **dcal** command may be used to convert decimal values to hexadecimal.

Machine status subcommands

stat	Display status information for a dump file.
switch [[tslot|eaddr] | [u|k]]	Switch the context to a particular thread or between user and kernel address space.
symptom [**-e**]	Display the symptom string for a dump. It is not valid on a running system.

Loader subcommands

lke [**?**] [**-l**] [slot|symb|eaddr]	Display information on loaded extensions.
stbl [*slot*|*symb*|*eaddr*]	Display loaded symbol tables.
rmst *slot*|*symb*|*eaddr*	Remove a symbol table.
exp [*symb*]	The **exp** command may be used to look for an exported symbol address or to display the export list.

Process subcommands

ppda [*|*cpu*|*symb*|*eaddr*]	Display per processor data areas.
intr [*slot*|*symb*|*eaddr*]	Display interrupt handler tables.
mst [*tslot*|*symb*|*eaddr*]	Display machine state save areas (mst).
p [*| – | *slot*|*symb*|*eaddr*]	Display process table entries.
th [*| – | *slot*|*symb*| *eaddr*|**-w state**]	Display thread table entries.

ttid [*tid*] Display detailed data for a thread table entry.
tpid [*pid*] Display summary information about each thread associated with a specified process.
rq [*bucket*|*symb*|*eaddr*] Display run queues.
sq [*bucket*|*symb*|*eaddr*] Display sleep queues.
lq [*bucket*|*symb*|*eaddr*] Display lock queues.
u [**-64**][*tslot*|*symb*|*eaddr*] Display the u-block for a thread.

LVM subcommands

pbuf [***] *symb*|*eaddr* Display physical buffer information.
volgrp *symb*|*eaddr* Display volume group information.
pvol *symb*|*eaddr* Display physical volume information.
lvol *symb*|*eaddr* Display logical volume information.

SCSI subcommands

ascsi [*slot*|*symb*|*eaddr*] Display SCSI adapter information.
vscsi [*slot*|*symb*|*eaddr*] Display virtual SCSI information.
scdisk [*slot*|*symb*| *eaddr*] Display SCSI disk information.

Memory allocation subcommands

hp [*symb*|*eaddr*] Display kernel heap information.
xm [**-?**] Display heap debug information.
kmbucket [**?**] [**-l**] [**-c** cpu] [**-i** index][*addr*] Display kernel memory allocator bucket information.
kmstats [*symb*|*eaddr*] Display kernel allocator memory statistics.

File system subcommands

buf [*slot*|*symb*|*eaddr*] Display buffer cache headers.
hb [*bucket*|*symb*|*eaddr*] Display the buffer cache hash list of headers.
fb [*bucket*|*symb*|*eaddr*] Display the buffer cache freelist of headers.
gno *symb*|*eaddr* Display data for a generic node structure.
gfs *symb*|*eaddr* Display data for a generic file system structure.
file [*slot*|*symb*|*eaddr*] Display the file table.
ino [*slot*|*symb*|*eaddr*] Display the inode table. Only used inodes are printed. Unused inodes may be printed with the **fino** subcommand.
hino [*bucket*|*symb*|*eaddr*] Display inode hash lists.
fino [*slot*|*symb*|*eaddr*] Display inode (used and unused) cache list entries.
rno *symb*|*eaddr* Display data for a remote node structure.
cku *symb*|*eaddr* Display data for a client kudp private structure.
vno *symb*|*eaddr* Display data for a virtual node structure.
vfs [*slot*|*symb*|*eaddr*] Display data from the virtual file system table.
specno *symb*|*eaddr* Display special device nodes data.
devno [*slot*|*symb*|*eaddr*] Display device node table information.
fifono [*slot*|*symb*|*eaddr*] Display fifo node table information.
hno [*bucket*|*symb*|*eaddr*] Display hash node table information.

System table subcommands

var Display the **var** structure and the system configuration of the machine.
dev [*symb*|*addr*|*major*] Display the device switch table.

trb	Display timer request blocks.
slk [*symb*|*eaddr*]	Display data for simple locks.
clk [*symb*|*eaddr*]	Display data for complex locks.
ipl [*|*cpu*]	Display processor information table data.
trace [**-h**] [*hook*[:*subhook*]]... [**#***data*]... [**-c***channel*]	Display data from kernel trace buffers.
trace -K [**-j** *event1, eventN* | **-k** *event1, eventN*]	Display data in the trace buffers collected through the **trcstart** subcommand.
trcstart -[**f** | **l**] **-j** *event1, eventN* **-k** *event1, eventN* **-p**	Start the system trace through **kdb**.
trcstop	Stop the system trace started through **kdb**.

Network subcommands

ifnet	Display interface information.
tcb [*slot*|*symb*|*eaddr*]	Display data for TCP blocks.
udb [*slot*|*symb*|*eaddr*]	Display data for UDP blocks.
sock [**tcp**|**udp**] [*symb*|*eaddr*]	Display socket information.
sockinfo [**socket** | **inpcb** | **unpcb** | **rawcb** | **ribcb** | **tcpcb**]	Display Socket Information by type and address.
tcpcb [**tcp**|**udp**] [*symb*|*eaddr*]	Display TCP control blocks.
mbuf [**tcp**|**udp**] [*symb*|*eaddr*]	Display TCP/UDP message buffers.
ndd [*symb*|*eaddr*]	Display network device driver statistics.
netstat [**-n**] [**-D**] [**-c**] [**-P**] [**-m** | **-s** | **-ss** | **-u** | **-v**] [{**-A -a**} | {**-r -C -i -I** Interface}] [**-f** AddressFamily] [**-p** Protocol] [**-Zc** | **-Zi** | **-Zm** | **-Zs**] [Interval] [System]	Display network status.

VMM subcommands

vmker	Display virtual memory kernel data.
rmap [*] [*slot*]	Display the real address range mapping table.
pfhdata	Display the virtual memory control variables.
vmstat	Display virtual memory statistics.
vmaddr	Display virtual memory control structure addresses.
pdt [*] [*slot*]	Display the paging device table.
scb [*selection*]	Displays VMM segment control blocks.
pft [*selection*]	Display VMM page frame table data.
pte [*selection*]	Display VMM page table entries.
pta [**?**]	Display VMM PTA segment information.
ste [**-p** *pid*]	Display segment table entry information for 64-bit processes.
sr64 [**-p** *pid*] [*esid* [*count*]]	Display segment registers for 64-bit processes.
segst64 [**-p** *pid*] [**-e** *esid*] [**-s** flag] [*fno*|*shm*	Display segment state information for 64-bit processes.
apt [*selection*]	Display alias page table information.
vmwait [*symb*|*eaddr*]	Display VMM wait status information.
ames [*selection*]	Display address map for a specified process.
zproc	Display information about the VMM zeroing kproc.
vmlog	Display the current VMM error log entry.
vrld	Display the VMM reload xlate table. This information is only used on SMP PowerPC machines, to prevent VMM reload dead-lock.
ipc [*selection*]	Display interprocess communication facility information.
lka [*slot*|*symb*|*eaddr*]	Display VMM lock anchor data.
lkh [*slot*|*symb*|*eaddr*]	Display VMM lock hash list entries.
lkw [*slot*|*symb*|*eaddr*]	Display information about VMM lock words.

vmdmap [*slot*|*symb*|*eaddr*]	Display VMM disk map information.
vmlocks	Display VMM spin locks.

Examples

The following examples demonstrate invocation options for the **kdb** command

1. To invoke the **kdb** command with the default system image and kernel image files, type:

   ```
   kdb
   ```

   The **kdb** program returns a (0)> prompt and waits for entry of a subcommand.
2. To invoke the **kdb** command using a dump file named /var/adm/ras/vmcore.0 and the UNIX kernel file named /unix, type:

   ```
   kdb /var/adm/ras/vmcore.0 /unix
   ```

   The **kdb** program returns a (0)> prompt and waits for entry of a subcommand.

The following examples demonstrate usage of selected **kdb** commands:

1. To run the deadlock analysis subcommand, type:

   ```
   (0)> dla
   ```
2. To display the third entry in the vfs table, type:

   ```
   (0)> vfs 3
   ```
3. To display a list of vnodes, type:

   ```
   (0)> vnode
   ```

Files

/usr/sbin/kdb	Contains the **kdb** command.
/dev/pmem	Default system image file
/unix	Default kernel file

Related Information

KDB Kernel Debugger and kdb Command in the *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

Subcommands for the KDB Kernel Debugger and kdb Command in the *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

Memory Overlay Detection System (MODS) in the *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

---

kdestroy Command

Purpose

Destroys a Kerberos credentials cache.

Syntax

kdestroy [**-q**] [**-c** *cache_name* | **-e** *expired_time*]

Description

The **kdestroy** command deletes a Kerberos credentials cache file.

If you specify the **-e** flag, the command checks all of the credentials cache files in the default cache directory (**/var/krb5/security/creds**) and deletes any file which contains only expired tickets, provided the tickets have been expired for the specified *expired_time*.

Flags

-c *cache_name*

Specifies the name of the credentials cache you want to destroy. The default credentials cache is destroyed if you do not specify a command flag.

If the KRB5CCNAME environment variable is set, its value is used to name the default credentials (ticket) cache.

This flag is mutually exclusive with the **-e** flag.

-e *expired_time*

Specifies that all credentials cache files containing expired tickets be deleted if the tickets have been expired at least as long as the *expired_time* value.

The *expired_time* is expressed as *nwndnhnmns*, where:

n represents a number

w represents weeks

d represents days

h represents hours

m represents minutes

s represents seconds

You must specify the *expired_time* components in this order but you can omit any component. For example, `4h5m` represents four hours and 5 minutes and `1w2h` represents 1 week and 2 hours. If you only specify a number, the default is hours.

-q

Suppress the beep when **kdestroy** fails to destroy the ticket.

Security

To delete a credentials cache, the user must be the owner of the file or must be a root (uid 0) user.

Examples

1. To delete the default credentials cache for the user, type:

   ```
   kdestroy
   ```

2. To delete all credentials cache with expired tickets older than one day, type:

   ```
   kdestroy -e 1d
   ```

Files

/usr/krb5/bin/kdestroy

/var/krb5/security/creds/krb5cc_[uid] default credentials cache ([uid] is the UID of the user.)

Related Information

The kinit command, klist command, and env command.

keyadd Command

Purpose

keyadd retrieves objects from the source keystore and adds them to the destination keystore.

Syntax

keyadd [**-S** *servicename*] **-l** *label* **-s** *source_keystore* [**-d** *destination_keystore*] [*username*]

Description

The **keyadd** command retrieves the objects named by label from the source keystore and adds them to the destination keystore. In a keystore, a user may have the private key, public key and the certificate stored using the same label. All objects matching a label are copied regardless of the object type. If an object with the same label already exists in the destination keystore, the command returns an error. This forces the user to explicitly remove an existing object instead of blindly destroying it.

Attention: Generally, there is no way to recover a destroyed object.

The **-S** option specifies which end-entity services and libraries to use while adding the objects from the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keydelete** will use the default service, which is **local**. It is an error to specify a servicename which does not have an entry in the **/usr/lib/security/ pki/ca.cfg** file.

The **-l** option must be specified. This label uniquely identifies an object in the keystore to be copied. The **-s** option must also be specified.

If the **-d** option is not given, the username's default keystore file will be used as the destination keystore The user's default keystore location is **/var/pki/security/keys/**<*username*>.

If no ***username*** is given, the currend user's username will be used. The user will be prompted for the password of the destination keystore and the source keystore. If the destination keystore does not exist, one will be created and the user will be asked to enter the destination keystore password again for confirmation.

Flags

-S *servicename* Specifies which service module to use.

-l *label* Specifies the label associated with the key to be added.

-s *source_keystore* Species the location of the source keystore.

-d *destination_keystore* Specifies the location of the destination keystore.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Security

This is a **setuid** command. In order to list the contents of a keystore the user must know the password of the private keystore.

`Root` and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they know the password to the keystore. A non-privileged user is only allowed to list the keystore that he owns.

Audit

This command records the following event information:

KEY_Add <*username*>

Examples

To copy a keystore object labeled as `label` from **/var/pki/security/keys/src.keystore** to **/var/pki/security/keys/dst.keystore**, enter:

```
$ keyadd -s /var/pki/security/keys/src.keystore -d /var/pki/
security/keys/dst.keystore -l label pkitest
```

Files

/usr/lib/security/pki/policy.cfg

/usr/lib/security/pki/ca.cfg

Related Information

The **certadd**, **certcreate**, **certdelete**, **certget**, **certlink**, **certlist**, **certrevoke**, **certverify**, **keydelete**, **keylist**, **keypasswd** and **mksecpki** commands.

---

keycomp Command

Purpose

Compiles a keyboard mapping file into an input method keymap file.

Syntax

keycomp <*Infile* **>***Outfile*

Description

The **keycomp** command reads a textual description of the keyboard from standard input and produces a binary file that maps the keys to standard output. The binary file is used by the Input Method to translate key strokes into character strings.

You can *bind* characters and strings to keys on a keyboard with specified combinations of *modifier keys* called keyboard states, or you can specify particular key and state combinations as unbound (return nothing). All input keys are represented by *keysyms*, which stand for the key symbols that are usually used in the AIXwindows environment to represent keyboard input.

Any combination of modifier keys is possible when you press a key on the keyboard, but usually the keys are mapped into a smaller set of states. This state mapping can be specified.

Keycomp Source File

The input file used by the **keycomp** command consists of one or more lines. The items on the line are separated by a space. Each line begins with a keysym or a hexadecimal value for a keysym. The

hexadecimal value represents keyboard input in the AIXwindows environment. Items following the keysym represent the binding for a particular combination of the Ctrl, Alt, Shift, Lock, and Alt Graphic keys.

An item can be one of the following:

- Character surrounded by single quotation marks
- String surrounded by double quotation marks
- Keysym allowing mapping to other keysyms
- **U** indicating that the entry is unbound

Hexadecimal (**\x***XX*), octal (**\o***OOO*), and decimal (**\d***DDD*) notations of a byte can be contained in character and string items.

Keyboard States

Modifier keys (Shift, Lock, Ctrl, Alt, and Alt Graphics keys) change the state of the keyboard. They are used to select one item from a line corresponding to the input keysym. A value that is a combination of bits, each bit corresponding to a modifier key, indicates the state of a keyboard. The modifier keys increase in significance in the following order: Shift, Lock, Ctrl, Alt, and Alt Graphic modifier keys.

The bit combination or state value of a keyboard is mapped to one item of a line. The mapping is defined by the line beginning with the `%M` control, which can contain only numbers. The first number after the `%M` control is the item number. The numbers that follow the first number represent keyboard states, and they are all mapped to the item. See “Examples”.

Flags

<*InFile*	Specifies a source file to be compiled by the **keycomp** command.
>*OutFile*	Specifies the name of the keymap file to be created.

Examples

1. The following is an example of a line for `XK_a` keysym input:

   ```
   XK_a'a'  XK_A  XK_A  XK_a  '\x01'  U  "hello"
   ```

   A , (comma) can, but need not, follow each item. Regardless of whether a comma follows an item, a space or tab must separate the items.

   Blank lines and lines beginning with the # character, except control statements, are ignored. All text between the # and the following line is ignored unless the # is part of a string enclosed in single or double quotation marks. Therefore, you can place comments at the end of a line that contains only a single item.
2. The following line shows that the keyboard states Ctrl, Ctrl+Shift, and Ctrl+Shift+Lock are all mapped to the third item:

   ```
   %M 3      4 5 7
   ```

Files

/usr/include/x11/keysymdef.h	Contains standard keysym definitions.
/usr/include/x11/aix_keysym.h	Contains unique keysym definitions.
/usr/bin/keycomp	Contains the **keycomp** command.
/usr/lib/nls/loc/*.imkeymap.src	Contains imkeymap source information.
/usr/lib/nls/loc/*.imkeymap	Maps a keysym/modifier to a string.

Related Information

The **IMInitializeKeymap** subroutine.

The Input Method Overview in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

The National Language Support Overview in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

---

keydelete Command

Purpose

Deletes an object (key, certificate, etc) identified by the label from a keystore. If the label is ALL, all objects are deleted.

Syntax

keydelete [**-S** *ServiceName*] **-l** *Label* [**-p** *PrivateKeystore*] [*UserName*]

Description

The **keydelete** command deletes an object (key, certificate, etc) identified by the *Label*. If the *Label* is ALL, all objects are deleted. The **-S** flag specifies which end-entity services and libraries to use while deleting the objects from the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keydelete** uses the default service, which is **local**. An error is returned if a *ServiceName* is specified which does not have an entry in the **/usr/ lib/security/pki/ca.cfg** file.

The **-l** flag must be specified. The *Label* is a variable length text string that is used to map a key in the keystore to the certificate which contains the matching public key. If the *Label* is ALL, all the objects in the keystore are deleted.

If the **-p** flag is not given, the username's default keystore file is used. The user's default keystore location is **/var/pki/security/keys/**<*UserName*>.

If no *UserName* is given, the current user's user name is used. The user is prompted for the password of the keystore.

Flags

-S *ServiceName*	Specifies which service module to use.
-l *Label*	Specifies the label associated with the key to be added.
-p *PrivateKeystore*	Species the location of the source destination keystore.

Arguments

username - Specifies the user whose key is going to be deleted.

Security

This is a privileged (set-UID root) command.

In order to list the contents of a keystore, the user must know the password of the private keystore.

root and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they know the password to the keystore. A non-privileged user is only allowed to list the keystore that he owns.

Audit
This command records the following event information:

KEY_Delete <*UserName*>

Examples
1. To delete a keystore object with a label **signcert** from the invoker's default keystore, type:
   ```
   keydelete -l signcert
   ```
2. To delete all the objects from the invoker's default keystore, type:
   ```
   keydelete -l ALL
   ```
3. To delete a keystore object with a label **signcert** from the keystore **/home/bob/ bob.keystore**, type:
   ```
   keydelete -p /home/bob/bob.keystore -l signcert
   ```

Files
/usr/lib/security/pki/ca.cfg

Related Information
The **keyadd**, **keylist**, and **keypasswd** commands.

---

keyenvoy Command

Purpose
Acts as an intermediary between user processes and the **keyserv** daemon.

Syntax
/usr/sbin/keyenvoy

Description
The **keyenvoy** command acts as an intermediary by some Remote Procedure Call (RPC) programs between their user processes and the **keyserv** daemon. An intermediary is necessary because the **keyserv** daemon talks only to root processes. This program cannot be run interactively.

Files

/usr/sbin/keyenvoy	Contains the **keyenvoy** command.

Related Information
The **keyserv** daemon.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference.

keylist Command

Purpose

keylist lists the keystore labels in a private keystore.

Syntax

keylist [**-S** *servicename*] [**-v** | **-c**] [**-p** *privatekeystore*] [*username*]

Description

The **keylist** command lists the keystore labels in a private keystore. The **-S** option specifies which end-entity services and libraries to use while listing the labels in the keystore. Available services are defined in **/usr/lib/security/pki/ca.cfg**. When invoked without **-S**, **keylist** will use the default service, which is **local**. It is an error to specify a servicename which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The user optionally may provide the location of the private keystore. If not given, the default location will be used. If the **-c** option is given, the type of the keystore object corresponding to the label will be specified by one letter symbol. The following are the symbols denoting the keystore object types:

P = Public Key

p = Private Key

T = Trusted Key

S = Secret Key

C = Certificate

t = Trusted Certificate

U = Useful Certificate

If the **-v** option is used, the type of the object for a label will be given in non-abbreviated version (for example, Public Key, Secret Key).

If required, the user will be prompted for the password of the underlying service keystore.

Flags

-S *servicename*	Specifies which service module to use.
-p *privatekeystore*	Specifies the location of the keystore.
-v	Specifies that the output is in verbose mode.
-c	Specifies a concise output.

Arguments

username	Specifies the AIX user whose key labels is going to be queried.

Exit Status

0	Successful completion.
>0	An error occured.

Security

This is a privileged (set-UID root) command.

In order to list the contents of a keystore the user must know the password of the private keystore.

`Root` and invokers belonging to group security are allowed to list anybody's keystore. However, they can only successfully complete this operation if they have the knowledge of the password to the keystore.

A non-privileged user is only allowed to list the keystore that he owns.

Audit

This command records the following event information:

KEY_List <*username*>

Examples

1. To list the labels in keystore **/var/security/pki/keys/bob**, enter:

   ```
   $ keylist -c -p /var/pki/security/keys/bob bob
   PpC label1
   PpC label2
   ```

2. To list labels/objects in verbose mode, enter:

   ```
   $ keylist -v -p /var/pki/security/keys/bob bob
   ```

Files

/usr/lib/security/pki/policy.cfg

/usr/lib/security/pki/ca.cfg

Related Information

The **certadd**, **certcreate**, **certdelete**, **certget**, **certlink**, **certlist**, **certrevoke**, **certverify**, **keyadd**, **keydelete**, **keypasswd** and **mksecpki** commands.

---

keylogin Command

Purpose

Decrypts and stores the user's secret key.

Syntax

/usr/bin/keylogin

Description

The **keylogin** command prompts users for their passwords. Then, the **keylogin** program decrypts the user's secret key, which is stored in the **/etc/publickey** file. The decrypted key is then stored by the local **keyserv** daemon to be used by any secure Remote Procedure Call (RPC) service, such as the Network File System (NFS).

The decrypted key given to the local **keyserv** daemon may eventually reach a time out and become invalid for that particular login session. The user can use the **keylogin** command again to refresh the key held by the **keyserv** daemon.

Files

/etc/publickey	Contains public or secret keys for NIS maps.

Related Information

The **chkey** command, **newkey** command.

The **keyserv** daemon.

How to Export a File System Using Secure NFS, How to Mount a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

List of NIS Commands.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

---

keylogout Command

Purpose

Deletes stored secret key.

Syntax

keylogout [**-f**]

Description

The **keylogout** command deletes the key stored by the key server process **keyserv**. Further access to the key is revoked; however, current session keys may remain valid until they expire or are refreshed.

Deleting the keys stored by **keyserv** will cause any background jobs or scheduled jobs that need secure RPC services to fail. Since only one copy of the key is kept on a machine, do not place a call to this command in your **logout** file since it will affect other sessions on the same machine.

Flags

-f	Forces **keylogout** to delete the secret key for the superuser. By default, **keylogout** by the superuser is disallowed because it would break all RPC services, such as NFS, that are started by the superuser.

Related Information

The **at** command, **chkey** command, **login** command, **keylogin** command, **newkey** command.

The **keyserv** daemon.

keypasswd Command

Purpose

keypasswd manages the passwords which are used to access a user's private keystore.

Syntax

keypasswd [**-S** *servicename*] [**-p** *privatekeystore* | **-k** *username*]

Description

The **keypasswd** command allows a user to change the password of a private keystore. The user will be asked to enter the old and new password of the keystore. The **-S** option specifies which end-entity services and libraries to use while changing the password. Available services are defined in the **/usr/lib/security/pki/ca.cfg** file. When invoked without **-S**, **keypasswd** will use the **local** service. You will get an error if you specify a servicename which does not have an entry in the **/usr/lib/security/pki/ca.cfg** file. The **-p** option specifies the private keystore for which the password is going to be changed. The **-k** option specifies the user's default private keystore. You will get an error if you specify both the **-k** and **-p** options.

Flags

-S *servicename* — Specifies which service module to use.

-p *privatekeystore* — Specifies the private keystore whose password is going to be changed.

-k — Specifies that the keystore to be used is that of username.

Security

This is a privileged (set-UID root) command.

To change the password of a keystore one must know the password of the keystore.

`Root` and invokers belonging to group security are allowed to change the password of any keystore as long as they know the password of the keystore. A non-privileged user is allowed to change only the keystore file that they own.

Audit

This command records the following event information:

KEY_Password <*username*>

Examples

1. To change the password of the default private keystore that is owned by Bob, enter:

   ```
   $ keypasswd
   ```

   where the invoker is Bob.
2. To change the password of any other private keystore, enter:

   ```
   $ keypasswd -p bob.keystore
   ```

Files

/usr/lib/security/ca.cfg

/usr/lib/security/policy.cfg

Related Information

The **certadd**, **certcreate**, **certdelete**, **certget**, **certlink**, **certlist**, **certrevoke**, **certverify**, **keyadd**, **keydelete**, **keylist**, and **mksecpki** commands.

---

keyserv Daemon

Purpose

Stores public and private keys.

Syntax

/usr/sbin/keyserv [**-n**]

Description

The **keyserv** daemon stores the private encryption keys of each user logged into the system. When a user types in a password during a **keylogin**, the secret key is decrypted. The decrypted key is then stored by the **keyserv** daemon. These decrypted keys enable the user to access secure network services such as secure Network File System (NFS).

When the **keyserv** daemon starts, it reads the key for the root directory from the **/etc/.rootkey** file. This daemon keeps the secure network services operating normally. For instance, after a power failure, when the system restarts itself, it gets the key for the root directory from the **/etc/.rootkey** file.

Flags

-n Prevents the **keyserv** daemon from reading the key for the root directory from the **/etc/.rootkey** file. Instead, the **keyserv** daemon prompts the user for the password to decrypt the root directory's key stored in the network information service map and then stores the decrypted key in the **/etc/.rootkey** file for future use. This option is useful if the **/etc/.rootkey** file ever goes out of date or is corrupted.

Examples

1. To start the **keyserv** daemon enabling the system to get the key for the root directory from the **/etc/.rootkey** file, enter:

   ```
   /usr/sbin/keyserv
   ```

2. A System Resource Controller (SRC) command can also enable the system to get the key for the root directory from the **/etc/.rootkey** file as follows:

   ```
   startsrc -s keyserv
   ```

   This command sequence starts a script that contains the keyserv daemon.

3. To prevent the **keyserv** daemon from reading the key for the root directory from the **/etc/rootkey** file, enter:

   ```
   chssys -s keyserv -a '-n'
   ```

   This command passes the **-n** argument to the **keyserv** daemon if SRC is used to start the daemon.

Files

/etc/.rootkey	Stores the encrypted key for the root directory.

Related Information

The **chssys** command, **keyenvoy** command, **startsrc** command.

How to Export a File System Using Secure NFS, How to Mount a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference.

System Resource Controller Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

kill Command

Purpose

Sends a signal to running processes.

Syntax

To Send Signal to Processes

kill [**-s** { *SignalName* | *SignalNumber* }] *ProcessID ...*

kill [**-** *SignalName* | **-** *SignalNumber*] *ProcessID ...*

To List Signal Names

kill -l [*ExitStatus*]

Description

The **kill** command sends a signal (by default, the **SIGTERM** signal) to a running process. This default action normally stops processes. If you want to stop a process, specify the process ID (PID) in the *ProcessID* variable. The shell reports the PID of each process that is running in the background (unless you start more than one process in a pipeline, in which case the shell reports the number of the last process). You can also use the **ps** command to find the process ID number of commands.

A root user can stop any process with the **kill** command. If you are not a root user, you must have initiated the process you want to stop.

SignalName is recognized in a case-independent fashion, without the SIG prefix.

If the specified *SignalNumber* is 0, the **kill** command checks the validity of the specified PID.

Flags

-s{*SignalName* **|** *SignalNumber***}**	Specifies the signal as a signal number or a signal name, such as **SIGKILL** -9 or **SIGTERM** -15.
-*SignalName*	Specifies a signal name, such as **SIGHUP**. This is obsolete syntax.
-*SignalNumber*	Specifies a signal number. This is obsolete syntax. **Note:** To specify the negative PID with the default signal in this syntax, you must specify **- -** as a signal. Otherwise the first operand is interpreted as a *SignalNumber*.
ProcessID	Specifies a decimal integer representing a process or process group to be signaled. If PID is a positive value, the **kill** command sends the process whose process ID is equal to the PID. If the PID value is 0, the **kill** command sends the signal to all processes having a process group ID equal to the process group ID of the sender. The signal is not sent to processes with a PID of 0 or 1. If the PID is -1, the **kill** command sends the signal to all processes owned by the effective user of the sender. The signal is not sent to processes with a PID of 0 or 1. If it is a negative number but not -1, the **kill** command sends the signal to all processes that have a process group ID equal to the absolute value of the PID.
-l	Lists all signal names supported by the implementation
-l*ExitStatus*	Lists signal names stripped of the common SIG prefix. If *ExitStatus* is an decimal integer value, the signal name corresponding to that signal is displayed. If *ExitStatus* is a value of the exit status corresponding to a process that was terminated by a signal, the signal name corresponding to the signal that terminated the process is displayed.

Exit Status

This command returns the following exit values:

0	At least one matching process was found for each *ProcessID* operand, and the specified signal was successfully processed for at least one matching process.
>0	An error occurred.

Examples

1. To stop a given process, enter:

   ```
   kill 1095
   ```

   This stops process `1095` by sending it the default **SIGTERM** signal. Note that process `1095` might not actually stop if it has made special arrangements to ignore or override the **SIGTERM** signal.
2. To stop several processes that ignore the default signal, enter:

   ```
   kill -kill 2098 1569
   ```

   This sends signal 9, the **SIGKILL** signal, to processes `2098` and `1569`. The **SIGKILL** signal is a special signal that normally cannot be ignored or overridden.
3. To stop all of your processes and log yourself off, enter:

   ```
   kill -kill 0
   ```

   This sends signal 9, the **SIGKILL** signal, to all processes having a process group ID equal to the senders process group ID. Because the shell cannot ignore the **SIGKILL** signal, this also stops the login shell and logs you off.
4. To stop all processes that you own, enter:

   ```
   kill -9 -1
   ```

This sends signal 9, the **SIGKILL** signal, to all processes owned by the effective user, even those started at other work stations and that belong to other process groups. If a listing that you requested is being printed, it is also stopped.

5. To send a different signal code to a process, enter:

```
kill  -USR1  1103
```

   The name of the **kill** command is misleading because many signals, including **SIGUSR1**, do not stop processes. The action taken on **SIGUSR1** is defined by the particular application you are running.

   Note: To send signal 15, the **SIGTERM** signal with this form of the **kill** command, you must explicitly specify -15 or **SIGTERM**.

Files

/usr/include/sys/signal.h	Specifies signal names.

Related Information

The **csh** command, **ksh** command, **ps** command, **sh** command.

The **kill** subroutine, **sigaction** subroutine.

---

killall Command

Purpose

Cancels all processes except the calling process.

Syntax

killall [**-**] [**-***Signal*]

Description

The **killall** command cancels all processes that you started, except those producing the **killall** process. This command provides a convenient means of canceling all processes created by the shell that you control. When started by a root user, the **killall** command cancels all cancellable processes except those processes that started it. If several Signals are specified, only the last one is effective.

If no signal is specified, the **killall** command sends a **SIGKILL** signal.

Flags

-	Sends a **SIGTERM** signal initially and then sends a **SIGKILL** signal to all processes that survive for 30 seconds after receipt of the signal first sent. This gives processes that catch the **SIGTERM** signal an opportunity to clean up. If both **-** and **-***Signal* are set, the **killall** command sends the specified signal initially and then sends a **SIGKILL** signal to all processes that survive for 30 seconds after receipt of the signal first sent.
-*Signal*	Sends the specified *Signal* number or *SignalName*.

Examples

1. To stop all background processes that have started, enter:

```
killall
```

   This sends all background processes the **kill** signal 9 (also called the **SIGKILL** signal).

2. To stop all background processes, giving them a chance to clean up, enter:
   ```
   killall -
   ```

   This sends signal 15, the **SIGTERM** signal; waits 30 seconds, and then sends signal 9, the **SIGKILL** signal.
3. To send a specific signal to the background processes, enter:
   ```
   killall -2
   ```

   This sends signal 2, the **SIGINT** signal, to the background processes.

Related Information

The **kill** command.

The **signal** subroutine.

---

kinit Command

Purpose

Obtains or renews the Kerberos ticket-granting ticket.

Syntax

kinit [**-l** *lifetime*] [**-r** *renewable_life*] [**-f**] [**-p**] [**-A**] [**-s** *start_time*] [**-S** *target_service*] [**-k** [**-t** *keytab_file*]] [**-R**] [**-v**] [**-c** *cachename*] [*principal*]

Description

The **kinit** command obtains or renews a Kerberos ticket-granting ticket. The Key Distribution Center (KDC) options specified by the [kdcdefault] and [realms] in the Kerberos configuration file (**kdc.conf**) are used if you do not specify a ticket flag on the command line.

If you are not renewing an existing ticket, the command reinitializes the credentials cache and will contain the new ticket-granting ticket received from the KDC. If you do not specify the *Principal* name on the command line and you do specify the **-s** flag, the *Principal* name is obtained from the credentials cache. The new credentials cache becomes the default cache unless you specify the cache name using the **-c** flag.

The ticket *Time* value for the **-l**, **-r** and **-s** flags is expressed as *ndnhnmns* where:

n represents a number

d represents days

h represents hours

m represents minutes

s represents seconds

You must specify the components in this order but you can omit any component, for example `4h5m` represents four hours and 5 minutes and `1d2s` represents 1 day and 2 seconds.

Flags

-A	Specifies that the ticket contain a list of client addresses. The ticket will contain the local host address list if this option is not specified. When an initial ticket contains an address list, it can be used only from one of the addresses in the the address list.
-c *cachename*	Specifies the name of the credentials cache to use. The default credentials cache is used if this flag is not specified. If the KRB5CCNAME environment variable is set, its value is used to name the default ticket cache. Any existing contents of the cache i are destroyed by **kinit**.
-f	Specifies that the ticket is to be forwardable. To forward the ticket, this flag must be specified.
-k	Specifies to obtain the key for the ticket principal from a key table. If you do not specify this flag, you are prompted to enter the password for the ticket principal.
-l *lifetime*	Specifies the ticket end time interval. The ticket cannot be used after the interval expires unless the ticket is renewed. The interval default time is 10 hours.
-p	Specifies that the ticket is to be proxiable. To make the ticket proxiable, this flag must be specified.
principal	Specifies the ticket principal. The principal is obtained from the credentials cache if the principal is not specified on the command line.
-r *renewable_life*	Specifies the renew time interval for a renewable ticket. The ticket cannot be renewed after the interval expires. The renew time must be greater than the end time. If this flag is not specified, the ticket is not renewable, although you can still generate a renewable ticket if the requested ticket lifetime exceeds the maximum ticket lifetime.
-R	Specifies to renew an existing ticket. No other flags may be specified when renewing an existing ticket.
-s *start_time*	Specifies a request for a postdated ticket, valid starting at *start_time*.
-S *target_service*	Specifies an alternate service name to use when getting initial tickets.
-t *keytab_file*	Specifies the key table name. The default key table is used if this flag is not specified and the **-k** flag is specified. The **-t** flag implies the **-k** flag.
-v	Specifies that the ticket granting ticket in the cache be passed to the kdc for validation. If the ticket is within its requested time range, the cache is replaced with the validated ticket.

Examples

1. To obtain a ticket-granting ticket with a lifetime of 10 hours, which is renewable for five days, type:
   ```
   kinit -l 10h -r 5d my_principal
   ```
2. To renew an existing ticket, type:
   ```
   kinit -R
   ```

Files

/usr/krb5/bin/kinit	
/var/krb5/security/creds/krb5cc_[uid]	default credentials cache ([uid] is the UID of the user.)

/etc/krb5/krb5.keytab	default location for the local host's keytab file.
/var/krb5/krb5kdc/kdc.conf	Kerberos KDC configuration file.

Related Information

The klist command, kdestroy command, and env command.

---

klist Command

Purpose

Displays the contents of a Kerberos credentials cache or key table.

Syntax

klist [[**-c**] [**-f**] [**-e**] [**-s**] [**-a**] [**-n**]] [**-k** [**-t**] [**-K**]] [*name*]

Description

The **klist** command displays the contents of a Kerberos credentials cache or key table.

Flags

-a	Displays all tickets in the credentials cache, including expired tickets. Expired tickets are not listed if this flag is not specified. This flag is valid only when listing a credentials cache.
-c	Lists the tickets in a credentials cache. This is the default if neither the **-c** nor the **-k** flag is specified. This flag is mutually exclusive with the **-k** flag.
-e	Displays the encryption type for the session key and the ticket.
-f	Displays the ticket flags using the following abbreviations:

F	Forwardable ticket
f	Forwarded ticket
P	Proxiable ticket
p	Proxy ticket
D	Postdateable ticket
d	Postdated ticket
R	Renewable ticket
I	Initial ticket
i	Invalid ticket
H	Hardware preauthentication used
A	Preauthentication used
O	Server can be a delegate

name	Specifies the name of the credentials cache or key table. The default credentials cache or key table is used if you do not specify a filename. If you do not specify a name indicating a cache name or keytab name, klist displays the credentials in the default credentials cache or keytab file as appropriate. If the KRB5CCNAME environment variable is set, its value is used to name the default credentials (ticket) cache.
-k	Lists the entries in a key table. This flag is mutually exclusive with the **-c** flag.
-K	Displays the encryption key value for each key table entry. This flag is valid only when listing a key table.
-n	Displays the numerical internet address instead of the host name. The default without the **-n** is host name. This command is used in conjunction with the **-a** flag.
-s	Suppresses command output but sets the exit status to 0 if a valid ticket-granting ticket is found in the credentials cache. This flag is valid only when listing a credentials cache.
-t	Displays timestamps for key table entries. This flag is valid only when listing a key table.

Examples

1. To list all of the entries in the default credentials cache, type:
   ```
   klist
   ```
2. To list all of the entries in the **etc/krb5/my_keytab** key table with timestamps, type:
   ```
   klist -t -k etc/krb5/my_keytab
   ```

Files

/usr/krb5/bin/klist	
/var/krb5/security/creds/krb5cc_[uid]	default credentials cache ([uid] is the UID of the user.)
/etc/krb5/krb5.keytab	default location for the local host's keytab file.

Related Information

The kinit command, kdestroy command, and env command.

---

kmodctrl Command

Purpose

Loads or unloads the kernel extension **/usr/lib/drivers/kmobip6**.

Syntax

kmodctrl [**-l**] [**-u**] [**-k**] [**-q**]

Description

The kernel extension **/usr/lib/drivers/kmobip6** contains support for the Mobile IPv6 functionality. This kernel extension must be loaded in order to configure the system as a mobile IPv6 home agent or

correspondent node. Normally this command will be run automatically by the **/etc/rc.mobip6** script if mobile IPv6 has been enabled using system management.

Flags

-k	Specifies an alternate path for the mobility kernel extension.
-l	Loads the mobility kernel extension.
-q	Checks whether the kernel extension is loaded.
-u	Unloads the mobility kernel extension.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example loads the kmobip6 kernel extension:

   ```
   kmodctrl -l
   ```

2. The following example unloads the kmobip6 kernel extension. This will disable all mobile IPv6 functionality on the system:

   ```
   kmodctrl -u
   ```

3. The following example queries whether the kmobip6 kernel extension is loaded:

   ```
   kmodctrl -q
   ```

Related Information

The **mobip6ctrl** command, **mobip6reqd** daemon, **ndpd-router** command, **rc.mobip6** command.

The Mobile IPv6 section in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

kpasswd Command

Purpose

Changes the password for a Kerberos principal.

Syntax

kpasswd [*Principal*]

Description

The **kpasswd** command changes the password for a specified Kerberos principal. It prompts for the current principals password, which is used to obtain a changepw ticket from the KDC for the user's Kerberos realm. If **kpasswd** successfully obtains the changepw ticket, the user is prompted twice for the new password and the password is changed.

If the principal is governed by a policy that specifies for example length and/or number of character classes required in the new password, the new password must conform to the policy.

You may not change the password for a ticket-granting service principal (krbtgt/domain) using the **kpasswd** command.

Parameters

Principal — Specifies the principal for which password you want to change. If you do not specify the principal on the command line, the principal is obtained from the default credentials cache.

Security

When requesting a password change, you must supply both the current password and the new password.

Files

/usr/krb5/bin/kpasswd

/var/krb5/security/creds/krb5cc_[uid] — default credentials cache ([uid] is the UID of the user.)

---

krlogind Daemon

Purpose

Provides the server function for the **rlogin** command.

Syntax

/usr/sbin/krlogind [-n] [-s]

> **Note:** The **krlogind** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

Description

The **/usr/sbin/krlogind** daemon is the server for the **rlogin** remote login command. The server provides a remote login facility.

Changes to the **krlogind** daemon can be made by using Web-based System Manager, the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the **/etc/inetd.conf** or **/etc/services** file. Entering **krlogind** at the command line is not recommended. The **krlogind** 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.

Service Request Protocol

When the **krlogind** daemon receives a service request, the daemon initiates the following protocol:

1. The **krlogind** daemon checks the source port number for the request. If the port number is not in the range 512 through 1023, the **krlogind** daemon terminates the connection.
2. The **krlogind** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **krlogind** daemon uses the dotted-decimal representation of the client host address.

3. The **krshd** daemon attempts to validate the user using the following steps:
   - makes sure that Kerberos 5 is a valid authentication method if the incoming ticket is a Kerberos 5 ticket. If the incoming ticket is a Kerberos 4 ticket, the connection fails. Kerberos 4 is not supported for **rlogin**.
   - calls **kvalid_user** with the local account name as well as the DCE principal.

Error Messages

The following error messages are associated with the **krlogind** daemon:

Try again	A fork command made by the server has failed.
/usr/bin/shell:	No shell. The shell specified for the shell variable cannot be started. The shell variable may also be a program.

Flags

-n Disables transport-level keep-alive messages. The messages are enabled by default.
-s Turns on socket level debugging.

Manipulating the krshd Daemon

The **krshd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **krshd** daemon is a member of the tcpip SRC subsystem group. Using the **chauthent** command will comment/uncomment the kshell line in the **/etc/inetd.conf** file and restart the **inetd** daemon depending on whether Kerberos 5 or Kerberos 4 is configured/unconfigured. This daemon should be manipulated using the **chauthent**/**lsauthent** commands. Direct modification of the **inetd.conf** file's kshell entry in not recommended.

Related Information

The **rlogin** command.

The **inetd** daemon, **rshd** daemon, **syslogd** daemon.

The **pty** special file.

The **kvalid_user** subroutine.

The **/etc/inetd.conf** file format.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Network Overview in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Secure Rcmds in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

krshd Daemon

Purpose

Provides the server function for remote command execution.

Syntax

/usr/sbin/krshd

Note: The **rshd** daemon is normally started by the **inetd** daemon. It can also be controlled from the command line, using SRC commands.

Description

The **/usr/sbin/krshd** daemon is the server for the **rcp** and **rsh** commands using Kerberos authentication. The **krshd** daemon provides remote execution of shell commands. These commands are based on requests from privileged sockets on trusted hosts. The shell commands must have user authentication. The **krshd** daemon listens at the kshell socket defined in the **/etc/services** file.

Changes to the **krshd** 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 **krshd** at the command line is not recommended. The **krshd** daemon is started by default when it is uncommented in the **/etc/inetd.conf** file.

The **inetd** daemon gets 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.

Service Request Protocol

When the **krshd** daemon receives a service request, it initiates the following protocol:

1. The **krshd** daemon checks the source port number for the request. If the port number is not in the range 0 through 1023, the **krshd** daemon terminates the connection.
2. The **krshd** daemon reads characters from the socket up to a null byte. The string read is interpreted as an ASCII number (base 10). If this number is nonzero, the **krshd** daemon interprets it as the port number of a secondary stream to be used as standard error. A second connection is created to the specified port on the client host. The source port on the local host is also in the range 0 through 1023.
3. The **krshd** daemon uses the source address of the initial connection request to determine the name of the client host. If the name cannot be determined, the **krshd** daemon uses the dotted decimal representation of the client host's address.
4. The **krshd** daemon retrieves the following information from the initial socket:
   - A Kerberos service ticket.
   - A null-terminated string of at most 16 bytes interpreted as the user name of the user on the client host.
   - Another null-terminated string interpreted as a command line to be passed to a shell on the local server host.
   - A null-terminated string of at most 16 bytes interpreted as the user name to be used on the local server host.
   - If the service ticket was a Kerberos 5 ticket, the daemon will expect either a Kerberos 5 TGT or a null string.
5. The krshd daemon attempts to validate the user using the following steps:
   - makes sure that Kerberos 5 is a valid authentication method if the incoming ticket is a Kerberos 5 ticket. Likewise, if the incoming ticket is a Kerberos 4 ticket, the Kerberos 4 authentication method must be configured.
   - calls **kvalid_user** with the local account name as well as the DCE Principal.
6. Once **krshd** validates the user, the **krshd** daemon returns a null byte on the initial connection. If the connection is a Kerberos 5 ticket and the TGT is sent, the command line passes to the **k5dcelogin** command, (which upgrades it to full DCE credentials). If the TGT is not sent or if the connection is a Kerberos 4 ticket, the command line passes to the user's local login shell. The shell then inherits the network connections established by the **krshd** daemon.

The **krshd** daemon is controlled by using the System Management Interface Tool (SMIT) or by changing the **/etc/inetd.conf** file. Entering **krshd** at the command line is not recommended.

Manipulating the krshd Daemon

The **krshd** daemon is a subserver of the **inetd** daemon, which is a subsystem of the System Resource Controller (SRC). The **krshd** daemon is a member of the tcpip SRC subsystem group. Using the **chauthent** command will comment/uncomment the kshell line in the **/etc/inetd.conf** file and restart the **inetd** daemon depending on whether Kerberos 5 or Kerberos 4 is configured/unconfigured. This daemon should be manipulated using the **chauthent**/**lsauthent** commands. Direct modification of the **inetd.conf** file's kshell entry in not recommended.

Related Information

The **rsh** command.

The **inetd** daemon.

The **kvalid_user** function.

The **/etc/hosts.equiv** file format, **/etc/inetd.conf** file format, and **/etc/services** file format.

Network Overview in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Secure Rcmds in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

ksh Command

Purpose

Invokes the Korn shell.

Syntax

ksh [**-i**] [{ **+** | **-** } { **a e f h k m n t u v x** }] [**-o** *Option ...*] [**-c** *String* | **-s** | **-r** | *File* [*Parameter*]]

Note: Preceding a flag with **+** (plus) rather than **-** (minus) turns off the flag.

Description

The **ksh** command invokes the Korn shell, which is an interactive command interpreter and a command programming language. The shell carries out commands either interactively from a terminal keyboard or from a file.

The Korn shell is backwardly compatible with the Bourne shell (invoked with the **bsh** command) and contains most of the Bourne shell features as well as several of the best features of the C shell.

For more information about the Korn shell, refer to ″Korn Shell″ in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Note: The **ksh** wait built in behaves in a manner similar to the **parent wait()** API.

An enchanced version of the korn shell, called ksh93, is also available. The enhanced Korn shell has additional features that are not available in the default Korn shell. For information regarding these additional features, refer to Enhanced Korn Shell in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Flags

-a	Exports automatically all subsequent parameters that are defined.
-c *String*	Causes the Korn shell to read commands from the *String* variable. This flag cannot be used with the **-s** flag or with the *File*[*Parameter*] parameter.
-e	Executes the **ERR** trap, if set, and exits if a command has a nonzero exit status. This mode is disabled while reading profiles.
-f	Disables file name substitution.
-h	Designates each command as a tracked alias when first encountered.
-i	Indicates that the shell is interactive. An interactive shell is also indicated if shell input and output are attached to a terminal (as determined by the **ioctl** subroutine). In this case, the **TERM** environment variable is ignored (so that the **kill 0** command does not kill an interactive shell) and the **INTR** signal is caught and ignored (so that a wait state can be interrupted). In all cases, the **QUIT** signal is ignored by the shell.
-k	Places all parameter assignment arguments in the environment for a command, not just those arguments that precede the command name.
-m	Runs background jobs in a separate process and prints a line upon completion. The exit status of background jobs is reported in a completion message. On systems with job control, this flag is turned on automatically for interactive shells.
-n	Reads commands and checks them for syntax errors, but does not execute them. This flag is ignored for interactive shells.

-o *Option* Prints the current option settings and an error message if you do not specify an argument. You can use this flag to enable any of the following options:

allexport
Same as the **-a** flag.

errexit Same as the **-e** flag.

bgnice Runs all background jobs at a lower priority. This is the default mode.

emacs Enters an emacs-style inline editor for command entry.

gmacs Enters a gmacs-style inline editor for command entry.

ignoreeof
Does not exit the shell when it encounters an end-of-file character. You must use the **exit** command, or override the flag and exit the shell by pressing the Ctrl-D key sequence more than 11 times.

keyword
Same as the **-k** flag.

markdirs
Appends a **/** (slash) to all directory names that are a result of filename substitution.

monitor
Same as the **-m** flag.

noclobber
Prevents redirection from truncating existing files. When you specify this option, use the redirection symbol **>|** (right caret, pipe symbol) to truncate a file.

noexec
Same as the **-n** flag.

noglob Same as the **-f** flag.

nolog Prevents function definitions from being saved in the history file.

nounset
Same as the **-u** flag.

privileged
Same as the **-p** flag.

verbose
Same as the **-v** flag.

trackall
Same as the **-h** flag.

vi Enters the insert mode of a vi-style inline editor for command entry. Entering escape character 033 puts the editor into the move mode. A return sends the line.

viraw Processes each character as it is typed in vi mode.

xtrace Same as the **-x** flag.

You can set more than one option on a single **ksh** command line.

-r Runs a restricted shell. With a restricted shell you cannot:

- Change the current working directory.
- Set the value of the **SHELL**, **ENV**, or **PATH** variable.
- Specify the pathname of a command that contains a **/** (slash).
- Redirect output of a command with **>** (right caret), **>|** (right caret, pipe symbol), **<>** (left caret, right caret), or **>>** (two right carets).

-s Causes the **ksh** command to read commands from the standard input. Shell output, except for the output of the special commands, is written to file descriptor 2. This parameter cannot be used with the **-c** flag or with the *File*[*Parameter*] parameter.

-t	Exits after reading and executing one command.
-u	Treats unset parameters as errors when substituting.
-v	Prints shell input lines as they are read.
-x	Prints executed commands and their arguments.

Files

/usr/bin/ksh	Contains the path name to the Korn shell.
/tmp/sh*	Contains temporary files that are created when a shell is opened.

Related Information

The **env** command.

The **profile** file format.

Korn Shell and Enhanced Korn Shell in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

kvno Command

Purpose

Displays the current key version number for a principal.

Syntax

kvno [**-e** *etype*] *service 1 service2*....

Description

The **kvno** command displays the current key version number for a principal (*service 1 service2*...). The security policy must allow a service ticket to be obtained for the principal. The current network identity is used when requesting the service ticket.

Flags

-e *etype*	Specifies which encryption *type* to get the current key version.
service 1 service2...	Specifies the principal for which you want to display the current key version number.

Security

The security policy must allow a service ticket to be obtained for the principal.

Files

/usr/krb5/bin/kvno

Related Information

The **klist** command.

last Command

Purpose

Displays information about previous logins.

Syntax

last [**-f** *FileName*] [**-T** *Time*] [**-n** *Number* | **-***Number*] [*Name ...*] [*Terminal ...*]

Description

The **last** command displays, in reverse chronological order, all previous logins and logoffs still recorded in the **/var/adm/wtmp** file. The **/var/adm/wtmp** file collects login and logout records as these events occur and holds them until the records are processed by the **acctcon1** and **acctcon2** commands as part of the daily reporting procedures. When the time daemon, timed, changes the system time, it logs entries in wtmp under the pseudo-user ″date″. An entry starting with ″date |″ is logged before the change, and one starting with ″date {″ is logged after the change. This allows for accurate accounting of logins that span a time change.

The list can be restricted to:

- The number of lines specified either with the **-***Number* parameter or with the **-n** flag.
- Logins or logoffs by the users specified by the *Name* parameter.
- Logins or logoffs from the terminals specified by the *Terminal* parameter.
- A terminal can be named fully or abbreviated as a tty. For example, you can specify either the `tty0` terminal or the `0` terminal.

> **Note:** If you specify both a *Name* and *Terminal* parameter, the **last** command displays all logins and logoffs meeting either criterion.

For each process, the **last** command displays the:

- Time the session began
- Duration
- Terminal (tty) used

If applicable, the following information is included:

- Terminations due to rebooting
- Sessions that are still continuing

If the **last** command is interrupted, it indicates how far the search has progressed in the **/var/adm/wtmp** file. If interrupted with a **quit** signal, the command indicates how far the search has progressed and then continues the search. The **quit** signal can be any one of the following:

```
#define SIGQUIT 3 /* (*) quit,
generated from terminal special char */

#define SIGKILL 9 /* kill (cannot be caught or ignored) */

#define SIGTERM 15 /* software termination signal */
```

The **kill** command sends the default SIGTERM signal when it is invoked without any option. If you want to send the SIGQUIT signal, enter the following:

```
kill -3 (Process ID)
```

See the **kill** command for more information.

Flags

-f *FileName* Specifies an alternate file from which to read logins and logoffs.
-n Specifies the number of lines to be displayed on the list.
-T *Time* Displays users logged in at the given Time value. 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)

.

Examples

1. To display all the recorded logins and logoffs by user `root` or from the `console` terminal, type:
   ```
   last root console
   ```
2. To display the time between reboots of the system, type:
   ```
   last reboot
   ```
   The `reboot` pseudo-user logs in when the system starts again.
3. To display all the users still logged in at 10.30 am on April 15th, enter:
   ```
   last -t 04151030
   ```
4. To display 10 lines in the list, type:
   ```
   last -n 10
   ```

Files

/usr/bin/last Contains the **last** command.
/var/adm/wtmp Contains connect-time accounting data, including login, logoff, and shutdown records.

Related Information

The **acctcon1** , **accton2** command, **lastlogin** command.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the Accounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Setting Up an Accounting System in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices* describes the steps you must take to establish an accounting system.

---

lastcomm Command

Purpose

Displays information about the last commands executed.

Syntax

lastcomm [*Command*] [*Name*] [*Terminal*]

Description

The **lastcomm** command displays information, in reverse chronological order, about all previously executed commands that are still recorded in the summary files in the **/var/adm/pacct** directory. You need to run the **/usr/sbin/acct/startup** command before you can execute the **lastcomm** command.

The list the **lastcomm** command displays can be restricted to:

- Commands specified by the *Command* parameter.
- Commands executed by the user specified by the *Name* parameter.
- Commands from the terminal specified by the *Terminal* parameter.

  A terminal can be named fully or abbreviated as a tty. For example, you can specify either the `tty0` terminal or the `0` terminal.

For each process, the following information is displayed:

- The name of the user who ran the process.
- Any flags the accounting facilities collected when the command executed. The following are valid flags:

S The root user executed the command.
F The command ran after a fork, but without a following subroutine.
C The command ran in PDP-11 compatibility mode.
D The command terminated with the generation of a core file.
X The command was terminated with a signal.

- The name of the command under which the process was called.
- The seconds of CPU time used by the process.
- The time the process was started.

Examples

1. To display information about all previously executed commands recorded in the **/var/adm/pacct** file, enter:

   ```
   lastcomm
   ```

2. To display information about commands named `a.out` executed by the `root` user on the `ttyd0` terminal, enter:

   ```
   lastcomm a.out root ttyd0
   ```

Files

/usr/bin/lastcomm	Contains the **lastcomm** command.
/var/adm/pacct	The directory that contains the current accounting summary files.

Related Information

The **acctcms** command.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the Accounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Setting Up an Accounting System in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices* describes the steps you must take to establish an accounting system.

lastlogin Command

Purpose
Reports the last login date for each user on the system.

Syntax
/usr/sbin/acct/lastlogin

Description
The **lastlogin** command updates the **/var/adm/acct/sum/loginlog** file to show the last date each user logged in. Normally, the **runacct** command, running under the **cron** daemon, calls this command and adds the information to the daily report. However, the **lastlogin** command can also be entered by a user who is a member of the ADM group.

> **Note:** You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

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.
/var/adm/acct/sum	Cumulative directory for daily accounting records.

Related Information
The **runacct** command.

The **cron** daemon.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the Accounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Setting Up an Accounting System in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices* explains the steps you must take to establish an accounting system.

lb_admin Command

Purpose
Administers the registration of NCS-based servers in location broker databases.

Syntax
lb_admin [**-nq**] [**-version**]

Description

The **lb_admin** tool administers the registrations of NCS-based servers in global location broker (GLB) or local location broker (LLB) databases. A server registers universal unique identifiers (UUIDs) specifying an

object, a type, and an interface, along with a socket address specifying its location. A client can locate servers by issuing lookup requests to GLBs and LLBs. The **lb_admin** tool can be used to look up information, add new entries, and delete existing entries in a specified database.

The **lb_admin** tool is useful for inspecting the contents of location broker databases and for correcting database errors. For example, if a server terminates abnormally without unregistering itself, use **lb_admin** to manually remove its entry from the GLB database.

When accepting input or displaying output, **lb_admin** uses either character strings or descriptive textual names to identify objects, types, and interfaces. A character string directly represents the data in a UUID in the format

xxxxxxxxxxxx.xx.xx.xx.xx.xx.xx.xx.xx

where each x is a hexadecimal digit. Descriptive textual names are associated with UUIDs in the **uuidname.txt** file.

The **lb_admin** command examines or modifies only one database at a time. This is referred to as the *current database*. The **use_broker** command selects the type of location broker database, GLB or LLB. The **set_broker** command selects the host whose GLB or LLB database is to be accessed. If one replica of a replicated GLB database is modified, the modifications are propagated to the other replicas of that database.

Flags

-nq Do not query for verification of wildcard expansions in unregister operations.
-version Display the version of NCS that this **lb_admin** belongs to, but do not start the tool.

Subcommands

In the **lookup**, **register**, and **unregister** commands, the *object*, *type*, and *interface* arguments can be either character strings representing UUIDs or textual names corresponding to UUIDs, as described earlier.

a[**dd**] Synonym for **register**.

c[**lean**]

Finds and deletes obsolete entries in the current database. When issuing this command, **lb_admin** attempts to contact each server registered in the database. If the server responds, the entry for its registration is left intact in the database. If the server does not respond, **lb_admin** tries to look up its registration in the LLB database at the host where the server is located, tells the result of this lookup, and asks if the entry is to be deleted. If a server responds, but its UUIDs do not match the entry in the database, **lb_admin** tells this result and asks if the entry is to be deleted.

There are two situations in which it is likely that a database entry should be deleted:

- The server does not respond. **lb_admin** succeeds in contacting the LLB at the host where the server is located, but the server is not registered with that LLB. The server is probably no longer running.
- Server responds, but its UUIDs do not match the entry in the database. The server that responded is not the one that registered the entry.

Entries that meet either of these conditions are probably safe to delete.

In other situations, it is best not to delete the entry unless it can be verified directly that the server is not running (for example, by listing the processes running on its host).

When **lb_admin** asks to delete an entry, there are four ways to respond. A **y**[**es**] response deletes the entry. A **n**[**o**] response leaves the entry intact in the database. After a yes or a no, **lb_admin** proceeds to check the next entry in the current database. A **g**[**o**] response invokes automatic deletion, in which all eligible entries are deleted and all ineligible entries are left intact, without the user being queried, until all entries have been checked. A **q**[**uit**] response terminates the clean operation.

d[**elete**]

Synonym for unregister.

h[**elp**] [*Command*] or ? [*Command*]

Displays a description of the specified *Command* or, if none is specified, list all of the **lb_admin** commands.

l[**ookup**] *Object Type Interface*

Looks up and displays all entries with matching *Object*, *Type*, and *Interface* fields in the current database. An asterisk can be used as a wildcard for any of the arguments. If all the arguments are wildcards, **lookup** displays the entire database.

q[**uit**]

Exits the **lb_admin** session.

r[**egister**] *Object Type Interface Location Annotation* [*Flag*]

Adds the specified entry to the current database. Use an asterisk to represent the nil UUID in the *Object*, *Type*, and *Interface* fields.

The location is a string in the format *Family:Host*[*Port*], where *Family* is an address family, *Host* is a host name, and *Port* is a port number. Possible values for *Family* include **ip**. A leading **#** can be used to indicate that a host name is in the standard numeric form. For example, `ip:vienna[1756]` and `ip:#192.5.5.5[1791]` are acceptable location specifiers.

The *Annotation* is a string of up to 64 characters annotating the entry. Use double quotation marks to delimit a string that contains a space or contains no characters. To embed a double quotation mark in the string, precede it with a backslash.

The *Flag* is either **local** (the default) or **global**, indicating whether the entry should be marked for local registration only or for registration in both the LLB and GLB databases. The *Flag* is a field that is stored with the entry but does not affect where the entry is registered. The **set_broker** and **use_broker** commands select the particular LLB or GLB database for registration.

s[**et_broker**] [*BrokerSwitch*] *Host*
Sets the host for the current LLB or GLB. If specifing global as the *BrokerSwitch*, **set_broker** sets the current GLB; otherwise, it sets the current LLB. The host is a string in the format *Family:Host*, where *Family* is an address family and *Host* is a host name. Possible values for *Family* include **ip**. A leading **#** can be used to indicate that a host name is in the standard numeric form. For example, `ip:prague` and `ip:#192.5.5.5` are acceptable host specifiers.

Issue **use_broker**, not this command, to determine if subsequent operations will access the LLB or the GLB.

set_t[**imeout**] [**short** | **long**]
Sets the timeout period used by **lb_admin** for all of its operations. With an argument of **short** or **long**, **set_timeout** sets the timeout accordingly. With no argument, it displays the current timeout value.

u[**nregister**] *Object Type Interface Location*
Deletes the specified entry from the current database.

The location is a string in the format *Family:Host*[*Port*], where *Family* is an address family, *Host* is a host name, and *Port* is a port number. Possible values for *Family* include **ip**. A leading **#** can be used to indicate that a host name is in the standard numeric form. For example, `ip:vienna[1756]` and `ip:#192.5.5.5[1791]` are acceptable location specifiers.

An asterisk can be used as a wildcard in the *Object*, *Type*, And *Interface* fields to match any value for the field. Unless queries have been suppressed by invoking **lb_admin** with the **-nq** option, **unregister** allows deletion of each matching entry. A **y**[**es**] response deletes the entry. A **n**[**o**] response leaves the entry in the database. A **g**[**o**] response deletes all remaining database entries that match, without querying. A **q**[**uit**] response terminates the **unregister** operation, without deleting any additional entries.

us[**e_broker**] [*BrokerSwitch*]
Selects the type of database that subsequent operations will access, GLB or LLB. The *BrokerSwitch* is either global or local. If a *BrokerSwitch* is not supplied, **use_broker** determines if the current database is global or local.

Use **set_broker** to select the host whose GLB or LLB is to be accessed.

Related Information

The **drm_admin** (NCS) command

The **glbd** (NCS) daemon, **llbd** (NCS) daemon, **nrglbd** (NCS) daemon.

---

lb_find Command

Purpose

Gets a list of global location broker (GLB) server daemons and their attributes.

Syntax

lb_find [**-q**] [**-v**] [**-dl**]

Description

The **lb_find** command sends out inquiries to the NCS location broker daemons and gathers the responses. The results are analyzed to determine whether the global location broker is replicatable, and which cell each daemon serves. After ten seconds, the results are summarized, showing the GLB broker type, the server host's network address, a cell name of either *default* or *alternate_N*, and the cell's UUID.

Flags

-q Queries for a GLB server, using the standard RPC mechanism. At most, one GLB server is printed, and only servers in the current machine's cell are searched. The program exits with a status of 0 if a GLB server is found; otherwise the status is nonzero.
-v Prints out the NCS version string.
-dl Turns on RPC debugging while searching for GLB servers.

Examples

A network contains one **glbd** in each of two NCS cells and one **nrglbd** in a third cell.

```
/etc/ncs/lb_find

sent to broadcast address 192.92.110.255

waiting for replies

received response from glb daemon at ip:stimpy(192.92.110.43)
port 1072.

received response from glb daemon at ip:oscar(192.92.110.16) port
1168.

received response from glb daemon at ip:vmess(192.92.110.21) port
1114.

.....

replicatable       ip:stimpy      default       333b91c50000.0d.0
0.00.87.84.00.00.00

replicatable       ip:oscar       alternate_1   54bdad9a4000.0d.0
0.01.83.0f.00.00.00

non_replicatable    ip:vmess      alternate_2   5c0e4acb8fa7.02.c
0.5c.6e.15.00.00.00
```

Related Information

The **lb_admin** command.

The **glbd** (NCS) daemon, **llbd** (NCS) daemon, **nrglbd** (NCS) daemon.

---

lbxproxy Command

Purpose

Low BandWidth X proxy.

Syntax

lbxproxy [**:**<*display*>] [**-help**] [**-display** *Display*] [**-motion** *Number*] [**-terminate** | **-reset**] [**-reconnect**] [**-l**] [**-nolbx**] [**-nocomp**] [**-nodelta**] [**-notags**] [**-nogfx**] [**-noimage**] [**-nosquish**] [**-nointernsc**] [**-noatomsfile**] [**-atomsfiles** *File*] [**-nowinattr**] [**-nograbcmap**] [**-norgbfile**] [**-rgbfile** *Path*] [**-tagcachesize**] [**-zlevel** *Level*] [**-compstats**] [**-nozeropad**] [**-cheaterrors**] [**-cheatevents**]

Description

The **lbxproxy** command accepts client connections, multiplexes them over a single connection to the X server, and performs various optimizations on the X protocol to make it faster over low bandwidth and/or high latency connections. Applications that would like to take advantage of the Low Bandwidth extension to

X (LBX) must make their connections to an **lbxproxy**. These applications need to know nothing about LBX, they simply connect to the **lbxproxy** as if were a regular server.

For authentication/authorization, **lbxproxy** passes the credentials presented by the client along to the server. Since X clients connect to **lbxproxy**, it is important that the user's **.Xauthority** file contain entries with valid keys associated with the network ID of the proxy. **lbxproxy** does not get involved with how these entries are added to the **.Xauthority** file. The user is responsible for setting it up.

The **lbxproxy** program has various flags, all of which are optional.

If **:<***Display***>** is specified, the proxy uses the *Display* port when listening for connections. The display port is an offset from port 6000, identical to the way in which regular X display connections are specified. If no port is specified on the command line, **lbxproxy** defaults to port 63. If the port that the proxy tries to listen on is in use, the proxy exits with an error message.

At startup, **lbxproxy** pre-interns a configurable list of atoms. This allows **lbxproxy** to intern a group of atoms in a single round trip and immediately store the results in its cache. While running, **lbxproxy** uses heuristics to decide when to delay sending window property data to the server. The heuristics depend on the size of the data, the name of the property, and whether a window manager is running through the same **lbxproxy**. Atom control is specified in the **AtomControl** file, set up during installation of **lbxproxy**, with command line overrides.

The file is a simple text file. There are three forms of lines: comments, length control, and name control. Lines starting with a **!** (exclamation point) are treated as comments. A line of the form z length specifies the minimum length in bytes before property data is delayed. A line of the form options atomname controls the given atom, where options is any combination of the following characters: **i** means the atom should be pre-interned; and **w** means data for properties with this name should be delayed only if a window manager is also running through the same **lbxproxy**.

Flags

-atomsfile *File*
Overrides the default **AtomControl** file.

-cheaterrors
Allows cheating on X protocol for the sake of improved performance. The X protocol guarantees that any replies, events or errors generated by a previous request are sent before those of a later request. This puts substantial restrictions on when **lbxproxy** can short circuit a request. The **-cheaterrors** flag allows **lbxproxy** to violate X protocol rules with respect to errors. Use at your own risk.

-cheatevents
The **-cheatevents** flag allows **lbxproxy** to violate X protocol rules with respect to events as well as errors. Use at your own risk.

-compstats
Reports stream compression statistics every time the proxy resets or receives a SIGHUP signal.

-display *Display*
Specifies the address of the X server supporting the LBX extension. If this flag is not specified, the display is obtained by the DISPLAY environment variable.

-help
Prints a brief help message about the command line flags.

-l
Causes all remaining arguments to be ignored.

-motion *Number*
Specifies the maximimum *Number* of events that can be in flight. A limited number of pointer motion events are allowed to be in flight between the server and the proxy at any given time. The default is 8.

-noatomsfile
Disables reading of the **AtomControl** file.

-nocomp
Disables stream compression.

-nodelta
Disables delta request substitutions.

-nogfx	Disables reencoding of graphics requests (not including image related requests).
-nograbcmap	Disables colormap grabbing.
-noimage	Disables image compression.
-nointernsc	Disables short circuiting of **InternAtom** requests.
-nolbx	Disables all LBX optimizations.
-norgbfile	Disables color name to RGB resolution in proxy.
-nosquish	Disables squishing of X events.
-notags	Disables usage of tags.
-nowinattr	Disables **GetWindowAttributes**/**GetGeometry** grouping into one round trip.
-nozeropad	Indicates to not zero out unused pad bytes in X requests, replies, and events.
-reconnect	Causes **lbxproxy** to reset (see -reset) and attempts to reconnect to the server when its connection to the server is broken. The default behavior of **lbxproxy** is to exit.
-rgbfile *Path*	Specifies an alternate RGB database *Path* for color name to RGB resolution.
-tagcachesize	Sets the size of the proxy's tag cache (in bytes).
-[terminate|reset]	The default behavior of **lbxproxy** is to continue running as usual when it's last client exits. The **-terminate** option will cause **lbxproxy** to exit when the last client exits. The **-reset** option will cause **lbxproxy** to reset itself when the last client exits. Resetting causes **lbxproxy** to clean up it's state and reconnect to the server.
-zlevel *Level*	Set the Zlib compression level (used for stream compression). The default is 9. 1 = worst compression, fastest. 9 = best compression, slowest.

ld Command

Purpose

Links object files.

Syntax

ld [**-D***Number*] [**-e***Label*] [**-G**] [**-H***Number*] [**-K**] [**-m**] [**-M**] [**-o***Name*] [**-r**] [**-s**] [**-S***Number*] [**-T***Number*] [**-u** *Name*] ... [**-v**] [**-z**] [**-Z***String*] ... [**-b***Option*] ... [**-L***Directory*] ... { **-f***FileID* ... **-l***Name* ... *InputFile* ... }

Description

The **ld** command, also called the linkage editor or binder, combines object files, archives, and import files into one output object file, resolving external references. It produces an executable object file that can be run. In addition, if you specify the **ld** command without the **-s** flag, you can use the output file as an *InputFile* parameter in another call to the **ld** command. By default, the **ld** command creates and places its output in the **a.out** file.

The **ld** command can relink a program without requiring that you list all input object files again. For example, if one object file from a large program has changed, you can relink the program by listing the new object file and the old program on the command line, along with any shared libraries required by the program. See "Examples" on page 177.

The **ld** command links input files in the order you specify on the command line. If you specify a file more than once, only the first occurrence of the file is processed. You must specify at least one input file, either

with the **-bI** (uppercase letter i), **-bimport**, **-bkeepfile**, **-f**, or **-l** (lowercase letter L) flag or as an *InputFile* parameter. (The **-bI**, **-bimport**, or **-bkeepfile** flag is the **-b** flag used with the **I**, **import**, or **keepfile** option.)

Use the **cc** command to link files when you are producing programs that run under the operating system. Because the **cc** command calls the **ld** command with common options and necessary support libraries, you do not need to specify them on the command line. (This information is read from the **/etc/xlC.cfg** or **/etc/vac.cfb** configuration file.)

Linking Mode

The **ld** command can link 32-bit objects and programs as well as 64-bit objects and programs, but 32-bit and 64-bit objects may not be linked together. To specify the mode for linking, you can use the **OBJECT_MODE** environment variable or the **-b32** or **-b64** options.

Archive Files

Archive files are composite objects, which usually contain import files and object files, including shared objects. If an archive file contains another archive file or a member whose type is not recognized, the **ld** command issues a warning and ignores the unrecognized member. If an object file contained in an archive file has the **F_LOADONLY** bit set in the XCOFF header, the **ld** command ignores the member. This bit is usually used to designate old versions of shared objects that remain in the archive file to allow existing applications to load and run. New applications link with the new version of the shared object, that is, another member of the archive.

Shared Objects

A shared object, usually created by another call to the **ld** command, is an object file with the **F_SHROBJ** bit set in the XCOFF header. A shared object defines external symbols that are resolved at run time. If you specify the **-bnso** or **-bnoautoimp** option, the **ld** command processes a shared object as an ordinary object file, and if the file is stripped, the link fails.

Ordinarily, a shared object used as input is only listed in the loader section of the output file if a symbol in the shared object is actually referenced. When the run-time linker is used, however, you might want shared objects to be listed even if there are no symbols referenced. When the **-brtl** option is used, all shared objects listed on the command-line that are not archive members are listed in the output file. The system loader loads all such shared objects when the program runs, and the symbols exported by these shared objects may be used by the run-time linker. Shared objects that are archive members are not loaded automatically unless automatic loading is enabled by an import file in the archive. To enable automatic loading, see “Import and Export File Format (-bI: and -bE: Flags)” on page 173.

Import and Export Files

Import files are ASCII files that identify the external symbols to resolve at run time. An import file identifies the shared object defining the imported symbols. The system loader finds and resolves those symbols at run time. If the first line of an import file begins with #! (#, exclamation point), you can specify the file on the command line as an ordinary *InputFile.* Otherwise, you must use the **-bI** or **-bimport** option to specify the import file.

Export files are ASCII files that identify external symbols that are made available for another executable object file to import. The file format of an export file is the same as the file format of an import file.

Libraries

Libraries are files whose names end in **.a**, or possibly **.so**. To designate a library, you can specify an absolute or relative path name or use the **-l** (lowercase letter L) flag in the form **-l***Name*. The last form designates a **lib***Name***.a** file, or in dynamic mode, a **lib***Name***.so** file, to be searched for in several directories. These search directories include any directories specified by **-L** flags and the standard library directories **/usr/lib** and **/lib**.

Note: If you specify a shared object, or an archive file containing a shared object, with an absolute or relative path name, instead of with the **-l***Name* flag, the path name is included in the import file ID string in the loader section of the output file. You can override this behavior with the **-bnoipath** option.

Processing

The **ld** command processes all input files in the same manner, whether they are archives or not. It includes the symbol tables of all objects, discarding only symbol definitions that duplicate existing symbols. Unlike some other versions of the **ld** command, you do not need to order archive files so references precede definitions. Furthermore, you do not need to list an archive file more than once on the command line.

The order of the **ld** command flags does not affect how they are processed, except for the flags used with input object files, libraries, and import files. These flags are: **-L**, **-f**, **-l** (lowercase letter L), **-bkeepfile**, and **-bI** (uppercase letter i). The flags are processed in the following order:

1. The **-L** flag adds a directory to the list of search directories to locate libraries specified by the **-l** (lowercase letter L) flag. The directories are searched in the order specified. All **-L** flags are processed before any **-l** flags are processed.
2. The **ld** command processes the *InputFile* parameters, the files specified by the **-f** flag and libraries specified by the **-l** (lowercase letter L) flag in the order specified.
3. The **ld** command processes import files specified by the **-bI** (uppercase letter i) flag in the order specified after processing all other object files and libraries. You can specify an import file as an input file without the **-bI** flag if it is necessary to process the file before processing some object files. In this case, the first line of the import file must begin with the #! (#, exclamation point) symbols, and the import file is processed with other input files as described in step 2.
4. The **-bkeepfile** option names an input file on which the **ld** command does not perform garbage collection. If the specified input file is also specified as an *InputFile* parameter or listed in a file specified by the **-f** flag, the **-bkeepfile** option does not affect the order in which the file is processed. Otherwise, the file is processed in order along with other input files, as described in step 2.

An output file produced by the **ld** command has execute permission set, unless you specify the **-r** flag or **-bnox** option or errors were reported while linking. An existing output file is not overwritten if any severe errors occurred, or if the output file was specified as an input file and any errors occurred.

Symbols

The **ld** command uses the following predefined symbols to provide special address locations and can be declared in C syntax as **extern char** *name***[ ]**.The symbol names are:

_text	Specifies the first location of the program.
_etext	Specifies the first location after the program.
_data	Specifies the first location of the data.
_edata	Specifies the first location after the initialized data
_end or **end**	Specifies the first location after all data.

The only way to use these symbols is to take their addresses. If an input file redefines any of these symbols, there may be unpredictable results. An additional predefined symbol, **_ptrgl**, is used by compilers to implement calls using function pointers.

Garbage Collection

By default, the **ld** command performs garbage collection, deleting control sections (CSECTs) that are not referenced when generating the output file.

A CSECT is an indivisible unit of coding or data. A CSECT references another CSECT if it contains a relocation entry (RLD) referring to a symbol contained in the other CSECT. A referenced CSECT causes

all CSECTs it references to be referenced as well. In addition, a CSECT is referenced if it contains exported symbols, symbols specified with the **-u** flag, or the symbol designated as the entry point with the **-e** flag.

If a symbol is not referenced but is needed in the output file, you can export the symbol, specify the symbol with the **-u** flag, or suppress garbage collection. To suppress garbage collection, use the **-r** flag or **-bnogc** option. To suppress garbage collection for individual object files, use the **-bkeepfile** option or the **-bgcbypass** option. Even when garbage collection is suppressed, unreferenced internal symbols are deleted.

Ignored and Unsupported Flags

For compatibility with other versions of the **ld** command, some flags are recognized but ignored. These flags produce a message stating that the flag and its operand were ignored. An ignored flag does not cause the **ld** command to stop without further processing. The following flags are ignored:

-ANumber	**-bnostrcmpct**	**-n**
-bfilelist	**-bstrcmpct**	**-N**
-bfl	**-B**Number	**-Q**
-bforceimp	**-d**	**-R**Number
-bi	**-i**	**-V**Number
-binsert	**-j[**Key**:]**Number	**-x**
-bnoforceimp	**-k**Key**:**Path	**-Y**Number

Flags that the **ld** command does not support result in an error message. After all unsupported flags are diagnosed, the **ld** command stops without further processing.

Flags

The ld command conforms to the XPG Utility Syntax Guidelines, except that the argument – only applies to the next operand, not to the remaining operands on the command line. For example, in the command line:

```
ld -- -s -v
```

The `-s` is treated as a filename and the `-v` is treated as a flag. To have `-v` treated as a filename, specify:

```
ld -- -s -- -v
```

Note: Enter a flag with an operand with or without a space between the flag and the operand. You can specify numeric values in decimal, octal (with a leading 0), or hexadecimal (with a leading 0x or 0X) format. If you specify conflicting flags on the command line, the **ld** command accepts the latest flag and ignores earlier ones.

-b*Option*
Sets special processing options. This flag can be repeated. For more information on these options, see “Options (-bOptions)” on page 165.

-D*Number*
Sets the starting address for the initialized data (the data section) of the output file to *Number.* If the specified number is -1, the data section starts immediately after the text section. By default, the data section begins at location 0.
Note: The system loader relocates the data section at run time, so the specified number only affects addresses listed in address maps or printed by utilities such as the **dump** or **nm** command.

-e*Label*
Sets the entry point of the executable output file to *Label*. The default entry point is **__start** (double underscore **start**).

-f*FileID*
Specifies a file containing a list of input files to process. FileID must contain a list of input file names. Each line in FileID is treated as if it were listed separately on the **ld** command line. Lines in the file can contain shell pattern characters * (asterisk), [(left bracket),] (right bracket), and ? (question mark), which are expanded using the **glob** subroutine and can designate multiple object files.

-G
Produces a shared object enabled for use with the run-time linker. The **-G** flag is equivalent to specifying the **erok**, **rtl**, **nortllib**, **nosymbolic**, **noautoexp**, and **M:SRE** options with the **-b** flag. Subsequent options can override these options.

-H*Number* Aligns the text, data, and loader sections of the output file so that each section begins on a file offset that is a multiple of *Number*. If the specified number is 1, no alignment occurs. If the specified number is 0, the loader section is aligned on a word boundary, and the text and data sections are aligned on a boundary so as to satisfy the alignment of all CSECTs in the sections. The default value is 0.

If the specified *Number* causes any CSECTS to be unaligned within the output file, the **ld** command issues a warning and the output executable file may not load or run.

-K Aligns the header, text, data, and loader sections of the output file so that each section begins on a page boundary. This flag is equivalent to specifying **-H***Number*, where *Number* is the page size of the machine on which **ld** is running.

-l*Name* (Lowercase L) In non-dynamic mode, processes the **lib***Name***.a** file. In dynamic mode, processes the **lib***Name***.so** or **lib***Name***.a** file. In all cases, directories specified by the **-L** flag or in the standard library directories (**/usr/lib** and **/lib**) are searched to find the file. In dynamic mode, the first directory containing either **lib***Name***.so** or **lib***Name***.a** satisfies the search. If both files are found in the same directory, **lib***Name***.so** is used. You can repeat this flag. For more information about dynamic mode, see "Run-time Linking" on page 173.
Note: The first definition of a symbol is kept, even if no reference to the symbol has been seen when the archive is read. In other versions of the **ld** command, a symbol defined in an archive is ignored if no reference to the symbol has been seen when the archive is read.

-L*Directory* Adds *Directory* to the list of search directories used for finding libraries designated by the **-l** (lowercase letter L) flag. The list of directories, including the standard library directories, is also recorded in the output object file loader section for use by the system loader unless you use the **-blibpath** or **-bnolibpath** option. You can repeat this flag.

-m or **-M** Lists to standard output the names of all files and archive members processed to create the output file. Shared objects and import files are not listed.

-o*Name* Names the output file *Name*. By default, the name of the output file is **a.out**.

-r Produces a nonexecutable output file to use as an input file in another **ld** command call. This file may also contain unresolved symbols. The **-r** flag is equivalent to specifying the **erok**, **noglink, nox**, and **nogc** options with the **-b** flag. (Subsequent options can override these options.)

-s Strips the symbol table, line number information, and relocation information when creating the output file. Stripping saves space but impairs the usefulness of the debuggers. You can also strip an existing executable by using the **strip** command.
Note: Non-shared objects cannot be linked if they are stripped. A shared object can be stripped, but a stripped, shared object cannot be used when linking statically.

-S*Number* Sets the maximum size (in bytes) allowed for the user stack when the output executable program is run. This value is saved in the auxiliary header and used by the system loader to set the soft ulimit. The default value is **0**.

For more information on large user stacks and 32-bit programs, see "Large Program Support Overview" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

-T*Number* Sets the starting address of the text section of the output file to *Number*. The default value is **0**.
Note: The system loader relocates the text section at run time, so the specified number affects only addresses listed in address maps or printed by utilities such as the **nm** or the **dump** command.

-u*Name* Prevents garbage collection of the external symbol *Name*. If the specified symbol does not exist, a warning is reported. You can repeat this flag.

-v Writes additional information about binder command execution to the loadmap file.

-z Functions the same as the **-K** flag.

-Z*String* Prefixes the names of the standard library directories with *String* when searching for libraries specified by the **-l** (lowercase letter L) flag. For example, with the **-Z/***test* and **-l***xyz* flags, the **ld** command looks for the **/test/usr/lib/libxyz.a** and **/test/lib/libxyz.a** files. When the **-Z***String* flag is used, the standard library directories are not searched. This flag has no effect on the library path information saved in the loader section of the output file. This flag is useful when developing a new version of a library. You can repeat this flag.

The Binder

The **ld** command verifies the command-line arguments and calls the binder (by default the **/usr/ccs/bin/bind** file), passing a generated list of binder subcommands. The binder program actually links the files. Although the binder is usually called by the **ld** command, you can start the binder directly. In this case, the binder reads commands from standard input.

Two options affect the calling of the binder. The **binder** option specifies which binder to call, and the **nobind** option prevents the **ld** command from calling a binder. Other binder options affect the binder subcommands that are generated.

If the **ld** command does not detect any errors in the options or command-line arguments, it calls the binder. The binder is called with a command line of the form:

```
bind [quiet_opt] [loadmap_opt]
```

The default value for *quiet_opt* is `quiet` and the default value for the *loadmap_opt* is the null string, so the default command line is:

```
/usr/ccs/bin/bind quiet
```

Options (-bOptions)

The following values are possible for the *Options* variable of the **-b** flag. You can list more than one option after the **-b** flag, separating them with a single blank.

Notes:

1. In the following list of binder options, two option names separated by the word *or* are synonymous.
2. The *FileID* indicates a path name. You can use either a relative or a full path name.
3. For a non-repeatable option that is followed by an argument, you can negate the option using a null argument. That is, specify only the option and the colon.
4. If you specify conflicting options, the last one takes precedence.

32 Specifies 32-bit linking mode. In this mode, all input object files must be XCOFF32 files, or an error is reported. XCOFF64 archive members are ignored. For import or export files specifying the mode of certain symbols, 64-bit symbols are ignored. If both **-b32** and **-b64** options are specified, the last specified option is used. If neither option is specified, the mode is determined from the value of environment variable **OBJECT_MODE**.

64 Specifies 64-bit linking mode. In this mode, all input object files must be XCOFF64 files, or an error will be reported. XCOFF32 archive members are ignored. For import or export files specifying the mode of certain symbols, 32-bit symbols are ignored. If both **-b32** and **-b64** options are specified, the last specified option is used. If neither option is specified, the mode is determined from the value of environment variable **OBJECT_MODE**.

asis Processes all external symbols in mixed case. This is the default. To process all external symbols in uppercase, see the **caps** option that follows.

autoexp Automatically exports some symbols from the output module without having to list them in an export file. (This option does not export all symbols from the output module. Use the **-bexpall** option to export all symbols.)This is the default. Use this option when linking a main program. The linker assumes that you are linking a main program when you do not specify a module type (with the **M** or **modtype** option) beginning with `S` and you do not use the **noentry** option.

When you use the **autoexp** option, if any shared object listed on the command-line imports a symbol from the special file **.** (dot), and the module being linked contains a local definition of the symbol, the symbol is exported automatically.

Other symbols are also exported automatically when you link with the **rtl** option. If a symbol defined in the module being linked has one or more additional definitions exported from a shared object listed on the command-line, and if any of the definitions is a BSS symbol, the symbol is exported automatically. If the definition in the module being linked is a BSS symbol, the symbol is exported with the `nosymbolic` attribute. Otherwise, the symbol is exported with the `symbolic` attribute. If the symbol is listed in an export file with another export attribute, the explicit attribute is used.

If the **autoexp** option would automatically export a symbol, but the symbol is listed in an export file with the **list** attribute, the symbol is not exported.

autoimp or **so** Imports symbols from any shared objects specified as input files. The shared objects are referenced but not included as part of the output object file. This is the default.

autoload: *path/file(member)* Automatically load archive member when the **-brtl** option is used.

bigtoc Generates extra code if the size of the table of contents (TOC) is greater than 64KB. Extra code is needed for every reference to a TOC symbol that cannot be addressed with a 16-bit offset. Because a program containing generated code may have poor performance, reduce the number of TOC entries needed by the program before using this option. The default is the **nobigtoc** option.

bindcmds:*FileID* Writes a copy of the binder commands generated by the **ld** command to *FileID*. You can redirect the resultant file as standard input to the binder program when the binder program is called as a standalone program. By default, no file is produced.

binder:*FileID* Uses *FileID* as the binder called by the **ld** command. The default binder is the **/usr/ccs/bin/bind** file.

bindopts:*FileID* Writes a copy of the binder program arguments to *FileID*. You can use the resultant file to start the binder program as a standalone program. By default, no file is produced.

C:*FileID* or **calls:***FileID* Writes an address map of the output object file to *FileID.* Symbols are sorted by section and then by address. For each symbol listed in the map, references from the symbol to other symbols are listed. By default, no file is produced. To learn more about the **calls** option, see “Address Maps” on page 176.

caps Processes all external symbols in uppercase. The default is the **asis** option.

comprld or **crld** Combines multiple relocation entries (RLDs) at the same address into a single RLD when possible. This is the default.

cror15 Uses the **cror 15,15,15** (`0x4def7b82`) instruction as the special no-op instruction following a call instruction. The default value is **ori 0, 0, 0** (`0x60000000`). See the **nop** option.

Use this option when linking object files on the current level of the system that you intend to relink on AIX 3.1.

cror31 Uses the **cror 31,31,31** (`0x4ffffb82`) instruction as the special no-op instruction following a call instruction. The default value is **ori 0, 0, 0** (`0x60000000`). See the **nop** option.

Use this option when linking object files on the current level of the system that you intend to relink on AIX 3.2.

D: *Number* [/*dsa*] or **maxdata:***Number*[/*dsa*]
Sets the maximum size (in bytes) allowed for the user data area (or user heap) when the executable program is run. This value is saved in the auxiliary header and used by the system loader to set the soft data ulimit. The default value is 0. When this option is used, the specified number of bytes are reserved for the user data area. The program may not explicitly map objects, using **shmat** or **mmap** functions, to virtual addresses that are reserved for the user data area.

For 32-bit programs, the maximum value allowed by the system is `0x80000000` for programs running under Large Program Support and `0xD0000000` for programs running under Very Large Program Support. See "Large Program Support Overview" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*. When a non-zero value is specified, the user data area begins in segment 3, and the program uses as many segments as neccessary to satisfy the **maxdata** value specified.

For 64-bit programs the **maxdata** option provides a guaranteed maximum size for the programs data heap. Any value can be specified but the data area cannot extend past `0x06FFFFFFFFFFFFF8` regardless of the maxdata value specified.

dbg:*Option* or **debugopt**:*Option*
Sets a special debugging or control option. By default, no debug option is set.

The **dbg:loadabs** or **debugopt:loadabs** option is used to indicate that the output program is loaded at the same address as the address specified by the **-T** and **-D** flags. In this case, a branch-absolute instruction is never changed to a (relative) branch instruction even if its target is a relocatable symbol. Similarly, a branch instruction is never changed to a branch-absolute instruction.

delcsect
Deletes all symbols in a CSECT if any symbol in the CSECT was defined by a previously read object file. This option prevents more than one instance of the same function from existing in the same program. For example, if **a.o** defines function **a()** and **b.o** defines functions **a()** and **b()**, linking **a.o** and **b.o** with the **-bdelcsect** option deletes symbols **a()** and **b()** from **b.o**. Thus, two instances of **a()** do not exist. The default is the **nodelcsect** option.

dynamic or **shared**
Cause the linker to process subsequent shared objects in dynamic mode. This is the default. In dynamic mode, shared objects are not statically included in the output file. Instead, the shared objects are listed in the loader section of the output file. When you specify the **rtl** option and dynamic mode is in effect, files ending in **.so** as well as **.a** satisfy searches for libraries specified with the **-l** (lowercase L) flag.

E:*FileID* or **export:***FileID*
Exports the external symbols listed in the file *FileID.* Exported symbols are listed in the loader section of the output file. There is no default export file.

ernotok or **f**
Reports an error if there are any unresolved external references. This is the default.

erok
Produces the output object file without errors even if there are unresolved external references. The default is the **ernotok** option.

errmsg
Writes error messages to standard error if the error level of the message is greater than or equal to the value of the **halt** option and the **quiet** option is used or standard output is redirected. This is the default.

ex1:*FileID*, **ex2:***FileID*, **ex3:***FileID*, **ex4:***FileID*, and **ex5:***FileID*
Provide user exits in the typical binder subcommand sequence. Each file specified by *FileID* must contain a list of binder subcommands, which will be run as follows:

ex1:*FileID*
Before reading any *InputFiles*

ex2:*FileID*
Immediately before symbol resolution

ex3:*FileID*
Immediately after symbol resolution

ex4:*FileID*
Immediately before writing the output file

ex5:*FileID*
Immediately after writing the output file

expall Exports all global symbols, except imported symbols, unreferenced symbols defined in archive members, and symbols beginning with an underscore (_). You can export additional symbols by listing them in an export file. This option does not affect symbols exported by the **autoexp** option.

When you use this option, you might be able to avoid using an export file. On the other hand, using an export file provides explicit control over which symbols are exported, and allows you to use other global symbols within your shared object without worrying about conflicting with names exported from other shared objects. The default is **noexpall**.

export:*FileID* Functions the same as the **E:***FileID* option.

f Functions the same as the **ernotok** option.

gc Performs garbage collection. Use the **nogc**, **gcbypass**, or **keepfile** option to prevent garbage collection for some or all object files. This is the default.

gcbypass:*Number* Specifies the number of files to bypass when garbage collecting if the **gc** option is specified. This option is ignored if the **nogc** option is used. If *Number* is 0, this option is equivalent to the **gc** option and garbage collection is performed for all files. The default value is **0**.

glink:*FileID* Uses the global linkage prototype code specified by *FileID.* Global-linkage interface code is generated for each imported or undefined function. In 32-bit mode, the default is the **/usr/lib/glink.o** file. In 64-bit mode, the default is the **/usr/lib/glink64.o** file.

h:*Number* or **halt:***Number* Specifies the maximum error level for binder command processing to continue. The default value is **4**. If any binder subcommand has a return value greater than *Number*, no additional binder subcommands are processed. If the halt level value is 8 or greater, the output file may not be executable if it is produced at all. Return values are:

0	No error
4	Warning
8	Error
12	Severe error
16	Internal program error

I:*FileID* or **import:***FileID* (Uppercase i) Imports the symbols listed in *FileID.* There is no default import file.

initfini:[*Initial*] [**:***Termination*] [**:***Priority*] Specifies a module initialization and termination function for a module, where *Initial* is an initialization routine, *Termination* is a termination routine, and *Priority* is a signed integer, with values from -2,147,483,648 to 2,147,483,647. You must specify at least one of *Initial* and *Termination*, and if you omit both *Termination* and *Priority*, you must omit the colon after *Initial* as well. If you do not specify *Priority*, 0 is the default. This option can be repeated.

This option sorts routines by priority, starting with the routine with the smallest (most negative) priority. It invokes initialization routines in order, and termination routines in reverse order.

This option invokes routines with the same priority in an unspecified order, but if multiple **initfini** options specify the same priority and both an initialization and termination routine, it preserves the relative order of the routines. For example, if you specify the options **initfini:i1:f1** and **initfini:i2:f2**, then function **i1** and **i2** are invoked in an unspecified order, but if **i1** is invoked before **i2** when the module is loaded, **f2** will be invoked before **f1** when the module is unloaded.
Note: IBM will only use priorities in the following inclusive ranges:

```
-2,147,483,640 to -2,147,000,000
-1,999,999,999 to -1,000,000,000
   -99,999,999 to -50,000,000
             0
    50,000,000 to 99,999,999
 1,000,000,000 to 1,999,999,999
 2,147,000,000 to 2,147,483,640
```

ipath For shared objects listed on the command-line, rather than specified with the **-l** flag, use the path component when listing the shared object in the loader section of the output file. This is the default.

keepfile:*FileID*
Prevents garbage collection of *FileID*. By default, the binder deletes unreferenced CSECTS in all files. You can repeat this option.

lazy
Enables lazy loading of a module's dependent modules. This option adds a **-lrtl** option following other flags and options. If the **-brtl** option is specified, the **-blazy** option is ignored and lazy loading is not enabled.

When a module is linked, a list of its dependent modules is saved in the module's loader section. The system loader automatically loads the dependent modules after the module is loaded. When lazy loading is enabled, loading is deferred for some dependents until a function is called in the module for the first time.

A module is lazy loaded when all references to the module are function calls. If variables in the module are referenced, the module is loaded in the typical way.
Note: Be careful while comparing function pointers if you are using lazy loading. Usually a function has a unique address to compare two function pointers to determine whether they refer to the same function. When using lazy loading to link a module, the address of a function in a lazy loaded module is not the same address computed by other modules. Programs that depend upon the comparison of function pointers should not use lazy loading.

For more information about lazy loading, refer to "Shared Libraries and Lazy Loading" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

l:*FileID* or **loadmap:***FileID*
(Lowercase L)Writes each binder subcommand and its results to *FileID.* By default, no file is produced.

libpath:*Path*
Uses *Path* as the library path when writing the loader section of the output file. *Path* is neither checked for validity nor used when searching for libraries specified by the **-l** flag. *Path* overrides any library paths generated when the **-L** flag is used.

If you do not specify any **-L** flags, or if you specify the **nolibpath** option, the default library path information is written in the loader section of the output file. The default library path information is the value of the **LIBPATH** environment variable if it is defined, and **/usr/lib:/lib**, otherwise.

loadmap:*FileID*
Functions the same as the **l:***FileID* option.

m:*UR*
Sets the **SGETUREGS** flag for the linker. When the **SGETUREGS** flag is set, the contents of the registers are stored in a buffer. This option is used by **coredump** system call.

M:*ModuleType* or **modtype:***ModuleType*
Sets the two-character module-type field and the shared object flag in the object file. The module type is not checked by the binder, but it should be set to one of the following values:

1L Single use. Module requires a private copy of the data section for each load.

RE Reusable. Module requires a private copy of the data area for each process dependent on the module.

RO Read only. Module is read-only, and can be used by several processes at one time.

If an **S** prefix is used on any of the preceding options, the shared flag in the object file is set. The system loader attempts to share a single instance of the data section of an RO module. Otherwise, the module type is ignored by the system loader. The default value is **1L**.

map:*FileID* or **R:***FileID*
Writes an address map of the output object file to *FileID.* Symbols are sorted by section and then by address. By default, no file is produced. To learn more about the **map** option, see "Address Maps" on page 176.

maxdata:*Number*[/*dsa*]
Functions the same as the **D:***Number*[/*dsa*] option.

maxstack:*Number* or **S:***Number*
Functions the same as the **-S** flag.

modtype:*ModuleType*
Functions the same as the **M:***ModuleType* option.

nl or **noloadmap**
Does not write the binder subcommands and their results to a load map file. This is the default.

noautoexp
Prevents automatic exportation of any symbols. The default is the **autoexp** option.

noautoimp or **nso**
Links any unstripped, shared objects as ordinary object files. When you use this option, the loader section of shared objects is not used. The default is the **autoimp** or **so** option.
Note: By using either of these flags, you statically link a shared object file into an application. Any application that is statically linked is *not* binary portable from any fix or release level to any other fix or release level.

nobigtoc
Generates a severe error message if the size of the TOC is greater than 64KB. If an output file is produced, it will not execute correctly. This is the default.

nobind
Omits calling the binder. Instead, the **ld** command writes the generated list of binder subcommands to standard output. By default, the **ld** command calls the binder.

nocomprld or **nocrld**
Does not combine multiple relocation entries (RLDs) at the same address into a single RLD. The default is the **comprld** or **crld** option.

nodelcsect
Allows all symbols in the CSECT to be considered during symbol resolution, even if some symbol in the CSECT is defined in a previously read object file. For more information, see the **delcsect** option. The **nodelcsect** option is the default.

noexpall
Does not export symbols unless you list them in an export file or you export them with the **autoexp** option. This is the default.

noentry
Indicates that the output file has no entry point. To retain any needed symbols, specify them with the **-u** flag or with an export file. You can also use the **-r** flag or the **nogc** or **gcbtpass** options to keep all external symbols in some or all object files. If neither the **noentry** nor the **nox** option is used and the entry point is not found, a warning is issued.

noerrmsg
Does not write error messages to standard error. Use this option if you specify the **noquiet** option and you pipe standard output to a command such as **tee** or **pg**.

nogc
Prevents garbage collection. CSECTs in all object files that contain global symbols are kept, whether they are referenced or not. The default is the **gc** option.

noglink
Prevents the **ld** command from inserting global linkage code. By default, the binder inserts the global linkage code.

noipath
For shared objects listed on the command-line, rather than specified with the **-l** flag, use a null path component when listing the shared object in the loader section of the output file. A null path component is always used for shared objects specified with the **-l** flag. This option does not affect the specification of a path component by using a line beginning with `#!` in an import file. The default is the **ipath** option.

nolibpath
Overrides any previous library path generated by the **-L** flag or specified by the **libpath** option. Instead, the default library path information is written in the loader section of the output file. The default library path information is the value of the **LIBPATH** environment variable if it is defined, and **/usr/lib:/lib** otherwise.

noloadmap
Functions the same as the **nl** option.

nom
Does not list the object files used to create the output file. This option overrides the **-m** flag. This is the default.

noobjreorder
Does not use the depth-first CSECT reordering logic. The CSECTs in the output file are arranged in the same order that the object files and library files were specified on the command line, except as follows:

- CSECTs are placed in their correct text, data, or BSS section of the object file, based on the storage-mapping class field of each CSECT.
- All CSECTs with a storage-mapping class of XMC_TC (TOC address constant) or XMC_TD (TOC variable) are grouped together.

If both the **noobjreorder** and **noreorder** options are specified, the **noreorder** option takes precedence. The default is the **reorder** option.

nop:*Nop*
Specifies the no-op instruction used after branches to local routines. *Nop* can be one of the special values **cror15**, **cror31**, **ori**, or an eight-digit hexadecimal number. The **ori** instruction is the default. Specifying the **-bnop:cror15** option is equivalent to specifying the **-bcror15** option; specifying the **-bnop:cror31** option is equivalent to specifying the **-bcror31** option. If you specify one of the special **nop** options, all previous **nop** options are overridden

If *Nop* is an eight-digit hexadecimal number, it specifies an arbitrary machine instruction. This machine instruction overrides any previously specified special value for *Nop* instruction. When you use this form, you can repeat this option.

The last machine instruction specified is the instruction generated by the binder after intramodule branches. Other specified machine instructions are recognized as no-op instructions, but are converted to the preferred no-op instruction.

noquiet
Writes each binder subcommand and its results to standard output. The default is the **quiet** option.

noreorder
Does not reorder CSECTs, except to combine all XMC_TC (TOC address constant) and XMC_TD (TOC variable) CSECTs and place them in the data section, and combine all BSS symbols and place them in the bss section. All other CSECTs are placed in the text section, so text and data are mixed in the output file. When the **noreorder** option is used, the text section of the output file may no longer be position-independent and the system loader will not load a module if the text section is not position-independent. Therefore, avoid using this option for programs and kernel extensions. If both **noobjreorder** and **noreorder** options are specified, the **noreorder** option takes precedence. The default is the **reorder** option.

nortl
Disables run-time linking for the output file. This option implies the **nortllib** and **nosymbolic-** options. Furthermore, additional actions described under the **rtl** option are not taken. This is the default.

nortllib
Does not include a reference to the run-time linker. If a main program is linked with this option, no run-time linking will take place in the program, regardless of the way any shared modules were linked that are used by the program. This is the default.

nostrip
Does not generate a stripped output file. Thus, the symbol table and relocation information is written in the output file. This option overrides the **-s** flag. This is the default.

nosymbolic
Assigns the **nosymbolic** attribute to most symbols exported without an explicit attribute. For more information, see “Attributes of Exported Symbols” on page 175. The default is the **nosymbolic-** option.

nosymbolic-
Assigns the **nosymbolic-** attribute to most symbols exported without an explicit attribute. For more information, see “Attributes of Exported Symbols.” This is the default.

notextro or **nro**
Does not check to ensure that there are no load time relocation entries for the text section of the output object file. This is the default.

notypchk
Does not check function-parameter types between external functional calls. The default is the **typchk** option.

nov
Does not write additional information to the load map file. This option is the default and overrides the **-v** flag.

nox
Does not make the output file executable. Neither the auxiliary header nor the loader section is written. Flags and options that specify values written in the auxiliary header or loader section have no effect when this option is used. The default is the **x** option.

nro
Functions the same as the **notextro** option.

nso
Functions the same as the **noautoimp** option.

pD:*Origin*
Specifies *Origin* as the address of the first byte of the file page containing the beginning of the data section. For example, if the data section begins at offset `0x22A0` in the object file, and `pD:0x20000000` is specified, the first byte of the data section is assigned address `0x200002A0`. This assumes a page size of 4096 (`0x1000`) bytes.

pT:*Origin*
Specifies *Origin* as the address of the first byte of the file page containing the beginning of the text section. For example, if the text section begins at offset `0x264` in the object file, and `pT:0x10000000` is specified, the first byte of the text section is assigned address `0x10000264`.

quiet
Does not write binder subcommands and their results to standard output. This is the default.

R:*FileID*
Functions the same as the **map:***FileID* option.

r or **reorder**
Reorders CSECTs as part of the **save** command processing. The reorder process arranges CSECTs of the same storage-mapping class by proximity of reference. This is the default.

rename:*Symbol, NewName*
Renames the external symbol *Symbol* to *NewName*. In effect, it is as if all definitions and references to *Symbol* in all object files were renamed to *NewName* before the files were processed. By default, symbols are not renamed.

reorder
Functions the same as the **r** option.

ro or **textro**
Ensures that there are no load time relocation entries for the text section of the resultant object file. The default is the **nro** option.

rtl
Enables run-time linking for the output file. This option implies the **rtllib** and **symbolic** options.

When dynamic mode is in effect (see the **dynamic** and **static** options), the **rtl** option allows input files specified with the **-l** flag to end in **.so** as well as in **.a**.

All input files that are shared objects are listed as dependents of your program in the output files loader section. The shared objects are listed in the same order as they were specified on the command line.

A shared object contained in an archive is only listed if the archive specifies automatic loading for the shared object member. You specify automatic loading for an archive member **foo.o** by creating a file with the following lines:

```
# autoload
#! (foo.o)
```

and adding the file as a member to the archive.You may also specify automatic loading for an archive member foo.o by using the **-bautoload** option:

```
-bautoload:<archive_name>(foo.o)
```

You may specify additional archive members with additional **-bautoloads**.

rtllib
Includes a reference to the run-time linker. The run-time linker is defined in **librtl.a**, and an implicit **-lrtl** flag is added automatically to the command line. This option (implied by the **rtl** option) must be used when linking a main program or no run-time linking will occur. Shared objects do not have to be linked with this option. The default is the **nortllib** option.

S:*Number*
Functions the same as the **-S** flag.

scalls:*FileID*
Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. For each symbol listed in the map, references from the symbol to the other symbols are listed. By default, no file is produced. To learn more about the **scalls** option, see “Address Maps” on page 176.

shared
Functions the same as the **dynamic** option.

smap:*FileID*
Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. By default, no file is produced. To learn more about the **smap** option, see “Address Maps” that follows.

so
Functions the same as the **autoimp** option.

stabcmpct:*Level* Specifies the level of compaction for stabstrings in the debug section. Stabstrings are strings that are longer than eight characters. Each substring in the symbol table has its own offset in the debug section. The following values are valid for *Level*:

0 Does not compact. Separate copies of duplicate stabstrings are written to the debug section.

1 Deletes duplicates. Each stabstring is written once to the `.debug` section. Duplicate stabstrings in the symbol table specifies the same offset into the debug section.

2 Renumbers the stabstrings and deletes most duplicates. (In some instances, multiple stabstrings can exist. They describe the same type but use different type numbers.) The scope of a type number is the entire output file, rather than a single input file as indicated by a C_FILE symbol table entry.

If the binder does not recognize a stabstring, it returns an error message and the resulting executable file does not have valid stabstrings. The rest of the file is unaffected by the error.

static Causes the linker to process subsequent shared objects in static mode. In static mode, shared objects are statically linked in the output file. Furthermore, files ending in **.so** are not found when searching for libraries specified with the **-l** flag.

sxref:*FileID* Writes an address map of the object file to *FileID*. Symbols are listed alphabetically. For each symbol listed in the map, references to the symbol from other symbols are listed. By default, no file is produced. To learn more about the **sxref** option, see "Address Maps." that follows.

symbolic Assigns the **symbolic** attribute to most symbols exported without an explicit attribute. For more information, see "Attributes of Exported Symbols" that follows. The default is the **nosymbolic-** option.

textro Same as the **ro** option.

typchk Performs function-parameter type checking between external functional calls. Parameter-type checking information can be included in object files by compilers and assemblers. This is the default.

For more information on type checking, see the "XCOFF (a.out) File Format" in *AIX 5L Version 5.2 Files Reference*.

x Makes the output file executable, if no errors exist. This is the default option.

X or **xref:***FileID* Writes an address map of the object file to *FileID*. Symbols are sorted by section and then by address. For each symbol listed in the map, references to the symbol from other symbols are listed. By default, no file is produced. To learn more about the **xref** option, see "Address Maps" that follows.

Run-time Linking

By default, references to symbols in shared objects are bound at link time. That is, the output module associates an imported symbol with a definition in a specific shared object. At load time, the definition in the specified shared object is used even if other shared objects export the same symbol.

You can cause your program to use the run-time linker, allowing some symbols to be rebound at load time. To create a program that uses the run-time linker, link the program with the **-brtl** option. The way that shared modules are linked affects the rebinding of symbols.

You can build shared objects enabled for run-time linking by using the **-G** flag. You can fully enable run-time linking for existing shared objects by relinking them with the **rtl_enable** command, as long as they have not been stripped.

Import and Export File Format (-bI: and -bE: Flags)

Each line within an import or export file must list a single symbol, followed by optional keywords or an optional address. Keywords are **svc**, **svc32**, **svc3264**, **svc64**, **syscall**, **syscall32**, **syscall3264**, **syscall64**, **symbolic**, **nosymbolic**, **nosymbolic-**, **list**, **cm**, **bss**, and **weak**. Only the keyword **weak** may be used in conjunction with another keyword.

In an import file, you can specify an address after the symbol to map data CSECTs to a shared memory segment and eliminate the need to use the assembler. You can also use one of the keywords **cm**, **bss**, or **weak** to specify the storage class of an imported symbol. When the **autoexp** option is used, the storage class of an imported symbol affects which symbols are automatically exported. If any other keyword is specified in an import file, the keyword is ignored.

In an export file, you can use the **svc** or **syscall** keyword after a name to indicate that it is a system call. This is needed when linking kernel extensions. You can use the **symbolic**, **nosymbolic**, or **nosymbolic-** keyword to associate an attribute with an exported symbol. For more information, see "Attributes of Exported Symbols" on page 175. You can use the **list** keyword to cause a symbol to be listed in the loader section of the output file, although it will not be marked as an exported symbol. This can be used for applications that want to process some symbols at run time. Listed symbols are not processed by the system loader or the run-time linker. A symbol address and the keywords **cm** and **bss** are ignored in an export file. The **weak** keyword can be used to specify weak symbol binding, and may be used with another attribute.

The **ld** command treats import and export files according to the following guidelines:

- A blank line is ignored.
- A line beginning with an * (asterisk) is a comment and is ignored.
- A line beginning with a # (#, blank space) provides operands to the **setopt** binder subcommand (**-bdbg:***Option*). For example, a line containing `# verbose` causes the binder to list each symbol as it is read from the file. These option settings are active only while processing the file. The # 32, # 64, # no32, and # no64 options can be used to specify whether the listed symbols should be used for 32-bit links, 64-bit links, or both.

  32-bit and 64-bit Import File Options

32	This option is used in an import or export file to specify that subsequent symbols should be processed when linking in 32-bit mode, but ignored when linking in 64-bit mode. If no **32** or **64** option is specified, all symbols are processed in both 32- and 64-bit modes.
64	This option is used in an import or export file to specify that subsequent symbols should be processed when linking in 64-bit mode, but ignored when linking in 32-bit mode. If no **32** or **64** option is specified, all symbols are processed in both 32- and 64-bit modes.
no32 or **no64**	Override a previous **32** or **64**. Subsequent symbols are processed in both 32- and 64-bit modes.

- When processing an import file, a line beginning with a #! (#, exclamation point) provides the shared library name to be associated with subsequent import symbols. The line can occur more than once and applies to subsequent symbols until the next line beginning with `#!` is read. This file name information is placed in the loader section of the XCOFF object file. It is used by the system loader to locate the appropriate object file at execution time. If the import file name is **ipath/ifile** (imember), the file name placed in the loader section is determined based on the import file name and the contents of the `#!` line of the import file, as follows:

#!	(Nothing after the `#!`) Use null path, null file, and null number. This is treated as a deferred import by the system loader.
#! ()	Use `ipath`, `ifile`, and `imember`. This line can be used if the import file is specified as an *InputFile* parameter on the command line. The file must begin with `#!` in this case. This line can also be used to restore the default name if it was changed by another `#!` line.
#! *path*/*file* **(***member***)**	Use the specified path, file, and member.
#! *path*/*file*	Use the specified path and file, and a null member.
#! *file*	Use a null path, the specified file, and a null member. At run time, a list of directories is searched to find the shared object.
#! (*member***)**	Use `ipath`, `ifile`, and the specified member. At run time, a list of directories is searched to find the shared object.
#! *file* **(***member***)**	Use a null path and specified file and member. At run time, a list of directories is searched to find the shared object.

#! . (A single dot) This name refers to the main executable. Use this file name when you are creating a shared object that imports symbols from multiple main programs with different names. The main program must export symbols imported by other modules, or loading will fail. This import file name can be used with or without the run-time linker.

#! .. (Two dots) Use this name to list symbols that will be resolved by the run-time linker. Use this file name to create shared objects that will be used by programs making use of the run-time linker. If you use a module that imports symbols from **..** in a program that was not linked with the **rtllib** option, symbols will be unresolved, and references to such symbols will result in undefined behavior.

To automatically load archive members when the **-brtl** option is used, you can create an import file as follows. If **shr.so** is a shared object in an archive, create an import file:

```
# autoload
#! (shr.so)
```

You can list additional member names on additional lines, if appropriate. You do not need to list symbol names in the import file because the symbols imported from **shr.so** will be read from **shr.so** itself.

For more information on creating a shared library, see "How to Create a Shared Library" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*. For more information on loading and binding, see the **load** subroutine in *AIX 5L Version 5.2 Technical Reference: Base Operating System and Extensions Volume 1*.

Attributes of Exported Symbols

Note: This section applies to AIX 4.2 or later.

When you use run-time linking, a reference to a symbol in the same module can only be rebound if the symbol is exported with the proper attribute. References to symbols with the **symbolic** attribute cannot be rebound. References to symbols with the **nosymbolic** attribute can be rebound. References to symbols with the **nosymbolic-** attribute can be rebound if the symbols are variables. For function symbols, calls using a function pointer can be rebound, while direct function calls cannot be rebound. The **nosymbolic-** attribute is the default and is provided for compatibility with previous versions of the operating system, but its use is not recommended.

If you are not using the run-time linker, *avoid* using the **nosymbolic** attribute because intra-module function calls will be made indirectly through a function descriptor using global-linkage code. Otherwise, the attribute of exported symbols has no effect for modules used with programs that do not use the run-time linker.

You can specify an explicit export attribute for symbols listed in an export file. Most symbols without an explicit attribute are exported with the default export attribute, as specified with the **symbolic**, **nosymbolic**, or **nosymbolic-** options.

The **weak** export attribute will mark the associated symbol's mapping type with L_WEAK in the loader section.

Imported symbols may only have the **weak** export attribute. If a symbol is imported from another module, all references to the symbol can be rebound. However, if a symbol is imported at a fixed address, all references are bound to this fixed address and cannot be rebound by the run-time linker. The system loader must resolve deferred imports. The run-time linker never resolves or rebinds references to deferred imports.

For exports of non-imported symbols, the following rules are used.

- If a symbol has the **list** attribute, it is listed in the loader section symbol table, but the **L_EXPORT** flag is not set in the symbol table entry. The run-time linker ignores such symbols.
- If a symbol was exported with an explicit attribute, the explicit attribute is used.
- If the symbol is a BSS symbol, it is exported with the **nosymbolic** attribute.
- Otherwise, the symbol is exported with the global attribute, as specified by the **symbolic**, **nosymbolic**, or **nosymbolic-** option. The default global attribute is **nosymbolic-**.

Address Maps

The **ld** command generates address maps, listing the layout of symbols in the output object file. If you use the **map** (or **R**) option, unresolved symbols and imported symbols are listed first, followed by the symbols in each section in address order. If you use the **calls** (or **C**) option, each symbol that is listed is followed by a list of references from that symbol to other symbols. If you use the **xref** (or **X**) option, each symbol that is listed is followed by a list of references to that symbol from other symbols. If you use the **smap**, **scalls**, or **sxref** option, the address map contains the same information as listed by the **map**, **calls**, or **xref** option, respectively, but symbols are listed in alphabetical order.

Internal symbols, with a storage class C_HIDEXT, are printed with the characters < and > (angle brackets) surrounding the symbol name. Names of external symbols, with a storage class C_EXT, are printed without the angle brackets, and those with a storage class of C_WEAKEXT, are printed with the characters { and } surrounding the symbol name.

Information listed about each symbol includes:

- An indication of whether the symbol is imported, exported, or the entry point. An * (asterisk) is used to mark the entry point, **I** is used to mark imported symbols, and **E** is used to mark exported symbols.
- Its address (except for imported symbols)
- Length and alignment (for CSECTs and BSS symbols)
- Storage-mapping class
- Symbol type
- Symbol number (used to differentiate between symbols of the same name)
- Symbol name
- Input file information

Storage-mapping classes and symbol types are defined in the **/usr/include/syms.h** file. In the address maps, only the last two characters are shown, except that storage-mapping class **XMC_TC0** is shown as **T0**.

The input file information depends on the type of input file. For object files, source files names obtained from C_FILE symbols table entries are listed. If the object is from an archive file, the object file name is listed in the following format:

ArchiveFileName[ObjectName]

A shared object name is listed between { } (braces). If a shared object is defined by an import file, the name of the import file is listed before the shared object name.

Import symbols have a symbol type of ER, but they have associated file input information. Undefined symbols are also listed with a symbol type of ER, but all other columns, except the symbol number, are left blank.

The **-T** and **-D** flags (or **pT** or **pD** options) affect the addresses printed in these address maps. For machine-level debugging, it is helpful to choose address so that symbols are listed with the same addresses that they have at run time. For a 32-bit program that does not use privately loaded shared objects, you can choose the proper addresses by specifying the `-bpT:0x10000000` and `-bpD:0x20000000` options. These options are defined by default in the **/etc/xlC.cfg** or **/etc/vac.cfg** file.

Environment Variables

The following environment variables affect the execution of the **ld** command:

LIBPATH	If **LIBPATH** is defined, its value is used as the default library path information. Otherwise, the default library path information is **/usr/lib:/lib**. If no **-L** flags are specified and no **-blibpath** option is specified, the default library path information is written in the loader section of the output file. Regardless of any options specified, **LIBPATH** is not used when searching for libraries that are specified from the command line.
TMPDIR	If the output file already exists or it is on a remote file system, the **ld** command generates a temporary output file. The temporary output file is created in the directory specified by **TMPDIR**. If **TMPDIR** is not defined, the temporary output file is created in the **/tmp** directory if the output file is remote, or in the same directory as the existing output file.
OBJECT_MODE	If neither the **-b32** nor **-b64** option is used, the **OBJECT_MODE** environment variable is examined to determine the linking mode. If the value of **OBJECT_MODE** is **32** or **64**, 32-bit or 64-bit mode is used, respectively. If the value is **32_64** or any other value, the linker prints an error message and exits with a non-zero return code. Otherwise, 32-bit mode is used.

Examples

1. To link several object files and produce an **a.out** file to run under the operating system, type:

   ```
   ld /usr/lib/crt0.o pgm.o subs1.o subs2.o -lc
   ```

   The `-lc` (lowercase letter L) links the **libc.a** library. A simpler way to accomplish this is to use the `cc` command (the compiler) to link the files as follows:

   ```
   cc  pgm.o  subs1.o  subs2.o
   ```

2. To specify the name of the output file, type:

   ```
   cc  -o pgm  pgm.o  subs1.o  subs2.o
   ```

   This creates the output in the file `pgm`.

3. To relink `pgm` if only the object file **subs1.o** has changed, type:

   ```
   cc -o pgm subs1.o pgm
   ```

   The CSECTs that originally came from object files **pgm.o** and **subs2.o** are read from the file **pgm**. This technique can speed the linking process if a program consists of many input files, but only a few files change at a time.

4. To link with library subroutines, type:

   ```
   cc  pgm.o  subs1.o  subs2.o  mylib.a  -ltools
   ```

   This links the object modules `pgm.o`, `subs1.o`, and `subs2.o`, the subroutines from the `mylib.a` archive, and the subroutine from the library specified by **-l** (lowercase letter L) flag. (This means the **/usr/lib/libtools.a** file).

5. To generate a shared object, type:

   ```
   ld -o shrsub.o subs1.o subs2.o -bE:shrsub.exp -bM:SRE -lc
   ```

   This links the object files **subs1.o**, **subs2.o**, and the subroutines from the library **libc.a** specified by **-lc** flag. It exports the symbols specified in the file **shrsub.exp** and stores the linked shared object in file **shrsub.o.** The `-bM:SRE` sets the shared object flag in the linked object file.

6. To link with the shared object `shrsub.o` generated previously, type:

   ```
   cc -o pgm pgm.o shrsub.o -L '.'
   ```

   This links the object file **pgm.o** with the exported symbols of `shrsub.o`. The linked output is stored in the object file `pgm`. The `-L '.'` adds the current directory to the library search path that the system loader uses to locate the `shrsub.o` shared object. At run time, this program is loaded only if it is run

from a directory containing an instance of the shrsub.o file or if the shrsub.o file is found in the **/usr/lib** standard library directory. To allow the program to be run from anywhere, use the option **-L `pwd`**.

The list of directories searched by the system loader can be seen using the **dump** command.

7. To link a program using the **libc.a** library as a non-shared library, type:

```
cc -o pgm pgm.o -bnso -bI:/lib/syscalls.exp
```

This links pgm.o with the necessary support libraries and names the output file pgm. For the cc command, the **libc.a** library is a necessary support library and is usually link-edited to the user's program as a shared library. In this example, the -bnso option directs the **ld** command to link with the **libc.a** library as a non-shared library, and the -bI:/lib/syscalls.exp directs the **ld** command to import the system call functions that are actually contained in the kernel or **/usr/lib/boot/unix** file. Whenever linking with the -bnso option, any symbols that were both imported and exported (that is, passed through) in a shared object must be explicitly imported, as is done by the -bI:/lib/syscalls.exp option in this example.

Note: Any time that **/usr/lib/libc.a** is linked non-shared, the flag **-bl:/lib/syscalls.exp** must be used. The application can also have to be linked again whenever an updated release of the operating system is installed. Any application that is statically linked is *not* binary portable from any fix or release level to any other fix or release level.

Files

/usr/bin/ld	Contains the **ld** command.
/usr/lib/lib*.a	Specifies libraries used for linking programs.
a.out	Specifies the default output file name.

Related Information

The **ar** command, **as** command, **nm** command, **dump** command, **rtl_enable** command, **strip** command.

The **a.out** file format.

The **glob** subroutine, **load** subroutine, **loadbind** subroutine, **loadquery** subroutine, **unload** subroutine.

"How to Create a Shared Library" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

"Large Program Support Overview" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

"Shared Libraries and Lazy Loading" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

"Shared Library Overview" in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

"XCOFF (a.out) File Format" in *AIX 5L Version 5.2 Files Reference*.

ldd Command

Purpose

Lists dynamic dependencies.

Syntax

ldd *FileName*

Description

The **ldd** command lists the path names of all dependencies. The command will report dependencies on only valid XCOFF files.

Parameters

FileName	Specifies the file whose dependencies will be listed.

Exit Status

0	The command completed successfully.
>0	An error occurred.

Examples

1. To display dependencies on **/usr/bin/dbx**, enter:

   ```
   ldd /usr/bin/dbx
   ```

   The output looks like the following:

   ```
   /usr/bin/dbx needs:
           /usr/lib/libc.a(shr.o)
           /usr/lib/libdbx.a(shr.o)
           /unix
           /usr/lib/libcrypt.a(shr.o)
           /usr/lib/libpthdebug.a(shr.o)
   ```

Files

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

Related Information

The **dump** command.

---

ldedit Command

Purpose

Modifies an **XCOFF** executable file header.

Syntax

ldedit -b *Option* [**-V**] *File*

Description

You can use the **ldedit** command to modify various fields in an **XCOFF** header or the auxiliary header of an executable file. The **ldedit** command makes it possible to mark or unmark an application as a 'large page data' program. The **ldedit** command also makes it possible to add or modify the values of MAXDATA and MAXSTACK without relinking.

The format of the **-b** flag is similar to the format used by the link editor, the **ld** command. The **-b** flag can be used multiple times on the command line.

If no flags are specified, the **ldedit** command dipslays a usage message using the standard error output.

Flags

-b *Option* Modifies an executable as specified by *Option*. The possible values for *Option* are:

lpdata Marks a file as a 'large page data' executable.

nolpdata
Unmarks a file as a 'large page data' executable.

maxdata:*value*
Sets the MAXDATA value. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

maxdata:*value***/dsa**
Sets the MAXDATA value and the DSA bit. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

maxstack:*value*
Sets the MAXSTACK value. *value* is an octal number when it starts with `0`, a hexadecimal number when it starts with `0x`, and a decimal number in all other cases.

-V Prints the version of the **ldedit** command on the standard error output.

Related Information

The **ld** command, **dump** command.

---

learn Command

Purpose

Provides computer-aided instruction for using files, editors, macros, and other features.

Syntax

learn[**-** *Directory*] [*Subject*[*LessonNumber*]]

Description

The **learn** command provides computer-aided instruction for using files, editors, macros, and other features. The first time you invoke the command, the system provides introductory information about the **learn** command. Otherwise, the **learn** command begins at the point where you left the last **learn** command session.

You can bypass the default action of the **learn** command by specifying the *Subject* parameter. The **learn** command starts with the first lesson of the subject you specify. You can specify any of the following subjects:

- Files
- Editors
- More files
- Macros
- EQN (the enquiry character)

- C (the language)

Note: You can only run the EQN lesson on a hardcopy terminal that is capable of 1/2 line motion. The **/usr/share/lib/learn/eqn/Init** file contains a detailed list of the supported terminals.

When you enter the **learn** command, the system searches the **/usr/share/lib/learn** directory for the appropriate lesson file. Use the **-***Directory* flag to identify a different search directory.

Subcommands

- The **bye** subcommand terminates a **learn** command session.
- The **where** subcommand tells you of your progress; the **where m** subcommand provides more detail.
- The **again** subcommand re-displays the text of the lesson.
- The **again** *LessonNumber* subcommand lets you review the lesson.
- The **hint** subcommand prints the last part of the lesson script used to evaluate a response; the **hint m** subcommand prints the entire lesson script.

Parameters

-*Directory*	Allows you to specify a different search directory. By default, the system searches for lesson files in the **/usr/share/lib/learn** directory.
LessonNumber	Identifies the number of the lesson.
Subject	Specifies the subject you want instruction on.

Examples

To take the online lesson about files, enter:

```
learn files
```

The system starts the **learn** program and displays instructions for how to use the program.

Files

/usr/share/lib/learn	Contains the file tree for all dependent directories and files.
/tmp/pl*	Contains the practice directories.
$HOME/.learnrc	Contains the startup information.

Related Information

The **ex** command.

Shells Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

leave Command

Purpose

Reminds you when you have to leave.

Syntax

leave [[**+**] *hhmm*]

Description

The **leave** command waits until the specified time and then reminds you that you have to leave. You are reminded at 5 minutes and at 1 minute before the actual time, again at that time, and at every minute thereafter. When you log off, the **leave** command exits just before it would have displayed the next message.

If you do not specify a time, the **leave** command prompts with `When do you have to leave?` A reply of `newline` causes the **leave** command to exit; otherwise, the reply is assumed to be a time. This form is suitable for inclusion in a **.login** or **.profile** file.

The **leave** command ignores interrupt, quit, and terminate operations. To clear the **leave** command, you should either log off or use the **kill-9** command and provide the process ID.

Flags

+ Specifies to set the alarm to go off in the indicated number of hours and minutes from the current time.

hhmm Specifies a time of day in hours and minutes (based on a 12- or 24-hour clock) or, if preceded by the **+**, a set number of hours and minutes from the current time for the alarm to go off. All times are converted to a 12-hour clock and assumed to relate to the next 12 hours.

Examples

To remind yourself to leave at 3:45, enter:

```
leave 345
```

Related Information

The **calendar** command.

---

lecstat Command

Purpose

Displays operational information about an Asynchronous Transfer Mode network protocol (ATM) Local Area Network (LAN) Emulation Client.

Syntax

lecstat [**-a -c -q -r -s -t -v**] *Device_Name*

Description

This command displays ATM LAN Emulation Client (LEC) operational information gathered by a specified LEC device. If a LEC device name is not entered, statistics for all available LEC's appear. Select a flag to narrow down your search results. You can display specific categories of information such as Configuration, LE_ARP Cache Entries, Virtual Connections, and Statistics, or you can choose to display all of the information categories.

You can also toggle debug tracing on or off and reset statistics counters.

Parameters

Device_Name The name of the LE Client device, for example, *ent1*.

Flags

-a	Requests all the LE Client information. This flag does not reset statistics counters or toggle trace.
-c	Requests the configuration.
-q	Requests the LE_ARP cache.
-r	Resets the statistics counters after reading.
-s	Requests the statistics counters.
-t	Toggles full debug trace on or off.
-v	Requests the list of virtual connections.

The following information appears for all valid calls and contains the following fields:

Device Type
Displays a description of the LAN Emulation Client (example: Ethernet or Token Ring)

LAN MAC Address
Displays the LAN Emulation Client's 6-byte Ethernet or Token Ring MAC address.

ATM Address
Displays the LAN Emulation Client's 20-byte Asynchronous Transfer Mode (ATM) address.

Elapsed Time
Displays the real time period which has elapsed since statistics were last reset.

Driver Flags
The current LAN Emulation Client(LEC) device driver NDD status flags. Example status flags:

```
Broadcast       Allowing broadcast packets.
Dead            Requires re-open.
Debug           Internal debug tracing enabled.
Limbo           Attempting ELAN recovery.
Running         Fully operational on the ELAN.
Up              Device has been opened.
```

Configuration Information
Selected with the **-a** or **-c** flags. Displays the network administrator's pre-configured attributes, as well as the current ELAN configuration values as defined by the LANE Servers.

Lane LE_ARP Table Entries
Selected with the **-a** or **-q** flags. Displays the current LE Client ARP cache. Included are the type of entry, it's state, the remote LAN MAC address or route descriptor, the remote ATM address and some descriptive values.

Example Types

```
BUS-PP              Broadcast and Unknown Server (point-to-point).
BUS-MP              Broadcast and Unknown Server (multi-point).
Data                Data (point-to-point).
LES-PP              LE Server (point-to-point).
LES-MP              LE Server (multi-point).
```

Example States

```
Arping          Attempting to locate remote client/server via LE_ARP.
Connected       Fully connected to the remote client/server.
Flushing        Flushing the data path to the client/server.
Known           Remote address is known but no connection yet.
Unknown         Remote address in unknown and not able to LE_ARP yet.
```

Lane Servers and Statistics
Selected with the **-a** or **-s** flags. Displays the current Transmit, Receive, and General statistics for this LE Client, as well as the ATM addresses of the current and available LANE Servers.

Lane connections

Selected with the **-a** or **-v** flags. Displays the current list of virtual connections in use by this LE Client. Included are virtual path and channel values, remote ATM address, and some descriptive values such as whether this connection was started by the remote, whether it is a duplicate connection, or whether the remote station is proxied by another LE Client.

Exit Status

If you specify an invalid *Device_Name*, this command produces error messages stating that it could not connect to the device. Examples of an invalid device error message might be:

```
LECSTAT: No LANE device configured.
LECSTAT: Device is not a LANE device.
LECSTAT: Device is not available.
```

Related Information

The atmstat command, entstat command, mpcstat command, and tokstat command.

---

lex Command

Purpose

Generates a C or C++ language program that matches patterns for simple lexical analysis of an input stream.

Syntax

lex [**-C**] [**-t**] [**-v**| **-n**] [*File*...]

Description

The **lex** command reads *File* or standard input, generates a C language program, and writes it to a file named **lex.yy.c**. This file, **lex.yy.c**, is a compilable C language program. A C++ compiler also can compile the output of the **lex** command. The **-C** flag renames the output file to **lex.yy.C** for the C++ compiler.

The C++ program generated by the **lex** command can use either STDIO or IOSTREAMS. If the cpp define _CPP_IOSTREAMS is true during a C++ compilation, the program uses IOSTREAMS for all I/O. Otherwise, STDIO is used.

The **lex** command uses rules and actions contained in *File* to generate a program, **lex.yy.c**, which can be compiled with the **cc** command. The compiled **lex.yy.c** can then receive input, break the input into the logical pieces defined by the rules in *File*, and run program fragments contained in the actions in *File*.

The generated program is a C language function called **yylex**. The **lex** command stores the **yylex** function in a file named **lex.yy.c**. You can use the **yylex** function alone to recognize simple one-word input, or you can use it with other C language programs to perform more difficult input analysis functions. For example, you can use the **lex** command to generate a program that simplifies an input stream before sending it to a parser program generated by the **yacc** command.

The **yylex** function analyzes the input stream using a program structure called a finite state machine. This structure allows the program to exist in only one state (or condition) at a time. There is a finite number of states allowed. The rules in *File* determine how the program moves from one state to another.

If you do not specify a *File*, the **lex** command reads standard input. It treats multiple files as a single file.

Note: Since the **lex** command uses fixed names for intermediate and output files, you can have only one program generated by **lex** in a given directory.

lex Specification File

The input file can contain three sections: *definitions*, *rules*, and *user subroutines*. Each section must be separated from the others by a line containing only the delimiter, %% (double percent signs). The format is:

```
definitions
%%
rules
%%
user subroutines
```

The purpose and format of each are described in the following sections.

Definitions

If you want to use variables in your rules, you must define them in this section. The variables make up the left column, and their definitions make up the right column. For example, if you want to define D as a numerical digit, you would write the following:

```
D   [0-9]
```

You can use a defined variable in the rules section by enclosing the variable name in {} (braces), for example:

```
{D}
```

Lines in the definitions section beginning with a blank or enclosed in %{, %} delimiter lines are copied to the **lex.yy.c** file. You can use this construct to declare C language variables to be used in the **lex** actions or to include header files, for example:

```
%{
#include <math.h>
int count;
%}
```

Such lines can also appear at the beginning of the rules section, immediately after the first %% delimiter, but they should not be used anywhere else in the rules section. If the line is in the definitions section of *File*, the **lex** command copies it to the external declarations section of the **lex.yy.c** file. If the line appears in the rules section, before the first rule, the **lex** command copies it to the local declaration section of the **yylex** subroutine in **lex.yy.c**. Such lines should not occur after the first rule.

The type of the **lex** external, **yytext**, can be set to either a null-terminated character array (default) or a pointer to a null-terminated character string by specifying one of the following in the definitions section:

```
%array    (default)
%pointer
```

In the definitions section, you can set table sizes for the resulting finite state machine. The default sizes are large enough for small programs. You may want to set larger sizes for more complex programs.

%a*n* Number of transitions is *n* (default 5000)
%e*n* Number of parse tree nodes is *n* (default 2000)
%h*n* Number of multibyte character output slots (default is 0)
%k*n* Number of packed character classes (default 1000)
%m*n* Number of multibyte ″character class″ character output slots (default is 0)
%n*n* Number of states is *n* (default 2500)
%o*n* Number of output slots (default 5000, minimum 257)
%p*n* Number of positions is *n* (default 5000)
%v*p* Percentage of slots vacant in the hash tables controlled by **%h** and **%m** (default 20, range 0 <= P < 100)
%z*n* Number of multibyte character class output slots (default 0)

If multibyte characters appear in extended regular expression strings, you may need to reset the output array size with the **%o** argument (possibly to array sizes in the range 10,000 to 20,000). This reset reflects the much larger number of characters relative to the number of single-byte characters.

If multibyte characters appear in extended regular expressions, you must set the multibyte hash table sizes with the **%h** and **%m** arguments to sizes greater than the total number of multibyte characters contained in the **lex** file.

If no multibyte characters appear in extended regular expressions but you want '.' to match multibyte characters, you must set **%z** greater than zero. Similarly, for inverse character classes (for example, [^abc]) to match multibyte characters, you must set both **%h** and **%m** greater than zero.

When using multibyte characters, the **lex.yy.c** file must be compiled with the **-qmbcs** compiler option.

Rules

Once you have defined your terms, you can write the rules section. It contains strings and expressions to be matched by the **yylex** subroutine, and C commands to execute when a match is made. This section is required, and it must be preceded by the delimiter %% (double percent signs), whether or not you have a definitions section. The **lex** command does not recognize your rules without this delimiter.

In this section, the left column contains the pattern in the form of an extended regular expression, which will be recognized in an input file to the **yylex** subroutine. The right column contains the C program fragment executed when that pattern is recognized, called an *action*.

When the lexical analyzer finds a match for the extended regular expression, the lexical analyzer executes the action associated with that extended regular expression.

Patterns can include extended characters. If multibyte locales are installed on your system, patterns can also include multibyte characters that are part of the installed code set.

The columns are separated by a tab or blanks. For example, if you want to search files for the keyword **KEY**, you can write the following:

```
(KEY) printf ("found KEY");
```

If you include this rule in *File*, the **yylex** lexical analyzer matches the pattern **KEY** and runs the **printf** subroutine.

Each pattern can have a corresponding action, that is, a C command to execute when the pattern is matched. Each statement must end with a **;** (semicolon). If you use more than one statement in an action, you must enclose all of them in { } (braces). A second delimiter, %%, must follow the rules section if you have a *user subroutine* section. Without a specified action for a pattern match, the lexical analyzer copies the input pattern to the output without changing it.

When the **yylex** lexical analyzer matches a string in the input stream, it copies the matched string to an external character array (or a pointer to a character string), **yytext**, before it executes any commands in the rules section. Similarly, the external int, **yyleng**, is set to the length of the matched string in bytes (therefore, multibyte characters will have a size greater than 1).

For information on how to form extended regular expressions, see ″Extended Regular Expressions in the lex Command″ in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

User Subroutines

The **lex** library defines the following subroutines as macros that you can use in the rules section of the **lex** specification file:

input	Reads a byte from **yyin**.
unput	Replaces a byte after it has been read.
output	Writes an output byte to **yyout**.
winput	Reads a multibyte character from **yyin**.
wunput	Replaces a multibyte character after it has been read.
woutput	Writes a multibyte output character to **yyout**.
yysetlocale	Calls the **setlocale (***LC_ALL***, " ");** subroutine to determine the current locale.

The **winput**, **wunput**, and **woutput** macros are defined to use the **yywinput**, **yywunput**, and **yywoutput** subroutines coded in the **lex.yy.c** file. For compatibility, these **yy** subroutines subsequently use the **input**, **unput**, and **output** subroutines to read, replace, and write the necessary number of bytes in a complete multibyte character.

You can override these macros by writing your own code for these routines in the user subroutines section. But if you write your own, you must undefine these macros in the definition section as follows:

```
%{
#undef input
#undef unput
#undef output
#undef winput
#undef wunput
#undef woutput
#undef yysetlocale
%}
```

There is no **main** subroutine in **lex.yy.c**, because the **lex** library contains the **main** subroutine that calls the **yylex** lexical analyzer, as well as the **yywrap** subroutine called by **yylex( )** at the end of *File*. Therefore, if you do not include **main( )**, **yywrap( )**, or both in the user subroutines section, when you compile **lex.yy.c**, you must enter `cc lex.yy.c -ll`, where `ll` calls the **lex** library.

External names generated by the **lex** command all begin with the preface **yy**, as in **yyin**, **yyout**, **yylex**, and **yytext**.

Finite State Machine

The default skeleton for the finite state machine is defined in **/usr/ccs/lib/lex/ncform**. The user can use a personally configured finite state machine by setting an environment variable **LEXER=***PATH.* The *PATH* variable designates the user-defined finite state machine path and file name. The **lex** command checks the environment for this variable and, if it is set, uses the supplied path.

Putting Blanks in an Expression

Normally, blanks or tabs end a rule and therefore, the expression that defines a rule. However, you can enclose the blanks or tab characters in " " (quotation marks) to include them in the expression. Use quotes around all blanks in expressions that are not already within sets of [ ] (brackets).

Other Special Characters

The **lex** program recognizes many of the normal C language special characters. These character sequences are:

Sequence	Meaning
\a	Alert
\b	Backspace
\f	Form Feed
\n	Newline (Do not use the actual new-line character in an expression.)
\r	Return
\t	Tab
\v	Vertical Tab
****	Backslash

Sequence	Meaning
\digits	The character with encoding represented by the one-, two-, or three-digit octal integer specified by **digits**.
\xdigits	The character with encoding represented by the sequence of hexadecimal characters specified by **digits**.
\c	Where **c** is none of the characters listed above, represents the character **c** unchanged.

Note: Do not use **\0** or **\x0** in **lex** rules.

When using these special characters in an expression, you do not need to enclose them in quotes. Every character, except these special characters and the operator symbols described in ″Extended Regular Expressions in the lex Command″ in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*, is always a text character.

Matching Rules

When more than one expression can match the current input, the **lex** command chooses the longest match first. When several rules match the same number of characters, the **lex** command chooses the rule that occurs first. For example, if the rules

```
integer    keyword action...;
[a-z]+       identifier action...;
```

are given in that order, and `integers` is the input word, **lex** matches the input as an identifier, because `[a-z]+` matches eight characters while `integer` matches only seven. However, if the input is integer, both rules match seven characters. **lex** selects the keyword rule because it occurs first. A shorter input, such as `int`, does not match the expression integer, and so **lex** selects the identifier rule.

Matching a String Using Wildcard Characters

Because **lex** chooses the longest match first, do not use rules containing expressions like `.*`. For example:

```
'.*'
```

might seem like a good way to recognize a string in single quotes. However, the lexical analyzer reads far ahead, looking for a distant single quote to complete the long match. If a lexical analyzer with such a rule gets the following input:

```
'first' quoted string here, 'second' here
```

it matches:

```
'first' quoted string here, 'second'
```

To find the smaller strings, `first` and `second`, use the following rule:

```
'[^'\n]*'
```

This rule stops after `'first'`.

Errors of this type are not far reaching, because the . (period) operator does not match a new-line character. Therefore, expressions like `.*` (period asterisk) stop on the current line. Do not try to defeat this with expressions like `[.\n]+`. The lexical analyzer tries to read the entire input file and an internal buffer overflow occurs.

Finding Strings within Strings

The **lex** program partitions the input stream and does not search for all possible matches of each expression. Each character is accounted for once and only once. For example, to count occurrences of both `she` and`he` in an input text, try the following rules:

```
she         s++
he          h++
\n          |.          ;
```

where the last two rules ignore everything besides he and she. However, because she includes he, **lex** does *not* recognize the instances of he that are included in she.

To override this choice, use the action **REJECT**. This directive tells **lex** to go to the next rule. **lex** then adjusts the position of the input pointer to where it was before the first rule was executed and executes the second choice rule. For example, to count the included instances of he, use the following rules:

```
she                 {s++;REJECT;}
he                  {h++;REJECT;}
\n                  |.                  ;
```

After counting the occurrences of she, **lex** rejects the input stream and then counts the occurrences of he. Because in this case she includes he but not vice versa, you can omit the **REJECT** action on he. In other cases, it may be difficult to determine which input characters are in both classes.

In general, **REJECT** is useful whenever the purpose of **lex** is not to partition the input stream but to detect all examples of some items in the input, and the instances of these items may overlap or include each other.

Flags

-C Produces the **lex.yy.C** file instead of **lex.yy.c** for use with a C++ compiler. To get the I/O Stream Library, use the macro, **_CPP_IOSTREAMS**, as well.

-n Suppresses the statistics summary. When you set your own table sizes for the finite state machine, the lex command automatically produces this summary if you do not select this flag.

-t Writes **lex.yy.c** to standard output instead of to a file.

-v Provides a one-line summary of the generated finite-state-machine statistics.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To draw **lex** instructions from the file lexcommands and place the output in **lex.yy.c**, use the following command:

   ```
   lex lexcommands
   ```

2. To create a **lex** program that converts uppercase to lowercase, removes blanks at the end of a line, and replaces multiple blanks by single blanks, including the following in a **lex** command file:

   ```
   %%
   [A-Z]   putchar(yytext[0]+ 'a'-'A');
   [ ]+$ ;
   [ ]+    putchar(' ');
   ```

Files

/usr/ccs/lib/libl.a	Contains the run-time library.
/usr/ccs/lib/lex/ncform	Defines a finite state machine.

Related Information

The **yacc** command.

Creating an Input Language with the lex and yacc Commands in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Using the lex Program with the yacc Program in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Example Program for the lex and yacc Programs in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

National Language Support Overview for Programming*AIX 5L Version 5.2 National Language Support Guide and Reference*.

---

libxrx Command

Purpose

RX Netscape Navigator Plug-in.

Description

The RX Plug-in may be used with Netscape Navigator (3.0 or later) to interpret documents in the `RX MIME` type format and start remote applications.

The RX Plug-in reads an RX document, from which it gets the list of services the application wants to use. Based on this information, the RX Plug-in sets the various requested services, including creating authorization keys if your X server supports the `SECURITY` extension. It then passes the relevant data, such as the X display name, to the application through an `HTTP GET` request of the associated CGI script. The Web server then executes the CGI script to start the application. The client runs on the web server host connected to your X server. In addition when the RX document is used within the `EMBED` tag (a Netscape extension to HTML), the RX Plug-in uses the `XC-APPGROUP` extension, if it is supported by your X server, to cause the remote application to be embedded within the browser page from which it was launched.

Installation

To install the RX Plug-in so that Netscape Navigator can use it, find the file named **libxrx.so** in **/usr/lib/X11** and copy it to either the system Netscape plugin directory or **$HOME/.netscape/plugins**. Do not install it as a symbolic link, this would confuse Netscape.

If you have configured Netscape Navigator to use the RX helper program (**xrx**), you must reconfigure it. Generally you simply need to remove or comment out the line you may have previously added in your mailcap file to use the RX helper program. Otherwise the plug-in will not be enabled. (The usual comment character for mailcap is `#`.) If you are already running Netscape Navigator, you need to exit and restart it after copying the plug-in library so the new plug-in will be found. Once this is done you can check that Navigator has successfully loaded the plug-in by checking the *About Plug-ins* page from the Help menu. This should show something like:

RX Plug-in

File name: /usr/local/lib/netscape/plugins/libxrx.sl.6.3

X Remote Activation Plug-in

Mime Type	Description	Suffixes	Enabled
application/x-rx	X Remote Activation Plug-in	xrx	Yes

Once correctly configured, Netscape Navigator will activate the RX Plug-in whenever you retrieve any document of the MIME type application/x-rx.

Resources

The RX Plug-in looks for resources associated with the widget **netscape.Navigator** (class **Netscape.TopLevelShell**) and understands the following resource names and classes:

xrxHasFirewallProxy (class XrxHasFirewallProxy)
Specifies whether an X server firewall proxy is running and should be used. Default is False.

xrxInternalWebServers (class XrxInternalWebServers)
The web servers for which the X server firewall proxy should not be used (only relevant when **xrxHasFirewallProxy** is ″True″). Its value is a comma separated list of mask/value pairs to be used to filter internal web servers, based on their address. The mask part specifies which segments of the address are to be considered and the value part specifies what the result should match. For instance the following list:

```
255.255.255.0/198.112.45.0,
255.255.255.0/198.112.46.0
```

matches the address sets: 198.112.45.* and 198.112.46.*. More precisely, the test is (address and mask) == value.

xrxFastWebServers (class XrxFastWebServers)
The web servers for which `LBX` should not be used. The resource value is a list of address mask/value pairs, as previously described.

xrxTrustedWebServers (class XrxTrustedWebServers)
The web servers from which remote applications should be run as trusted clients. The default is to run remote applications as untrusted clients. The resource value is a list of address mask/value pairs, as previously described.

Environment

If the RX document requests `X-UI-LBX` service and the default X server does not advertise the `LBX` extension, the RX Plug- in will look for the environment variable `XREALDISPLAY` to get a second address for your X server and look for the LBX extension there. When running your browser through lbxproxy you will need to set `XREALDISPLAY` to the actual address of your server if you wish remote applications to be able to use LBX across the Internet.

If the RX document requests `XPRINT` service, the RX Plug-in will look for the variables `XPRINTER`, `PDPRINTER`, `LPDEST`, `PRINTER`, and `XPSERVERLIST` to get the printer name and X Print server address to use. Note that although this set of variables allows to specify more than one server and printer, only the first pair will be used.

Finally, if you are using a firewall proxy, RX Plug-in will look for `PROXY_MANAGER` to get the address of your proxy manager (see **proxymngr**). When not specified it will use `:6500` as the default.

Related Information

The **xrx** command, **lbxproxy** command, and the **proxymngr** command.

---

line Command

Purpose

Reads one line from the standard input.

Syntax

line

Description

The **line** command copies one line from standard input and writes it to standard output. It returns an exit value of 1 on an end-of-file and always writes at least a new-line character. Use this command within a shell command file to read from the work station.

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 End-of-file occurred on input.

Examples

To read a line from the keyboard and append it to a file, create a script file as follows:

```
echo 'Enter comments for the log:'
echo ': \c'
line >>log
```

This shell procedure displays the message:

```
Enter comments for the log:
```

and then reads a line of text from the workstation keyboard and adds it to the end of the log. The `echo ': \c'` command displays a colon prompt. See the **echo** command for information about the **\c** escape sequence.

Related Information

The **echo** command, **ksh** command, **sh** command.

The **read** subroutine.

The Input and Output Handling Programmer's Overview in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs* describes the files, commands, and subroutines used for low-level, stream, terminal, and asynchronous I/O interfaces.

---

link Command

Purpose

Performs a **link** subroutine.

Syntax

link *File1 File2*

Description

The **link** command performs the **link** subroutine on a specified file. The **link** 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 **link** 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.

Note: If the . (dot) entry has been destroyed and the **fsck** command is unable to repair it (a rare occurrence), you can use the **link** command to restore the . (dot) entry of the damaged directory. Use the **link** *Dir Dir/*. command where the *Dir* parameter is the name of the damaged directory. However, use this only as a last resort when the directory is destroyed and the **fsck** command is unable to fix it.

Although the linked files and directories can be removed by the **unlink** command, it's safer to use the **rm** or **rmdir** command.

Examples

To create an additional link for an existing `file1`, enter:

```
link file1 file2
```

Files

/usr/sbin/link	Contains the **link** command.

Related Information

The **fsck** command, **unlink** command, **ln** command.

The **link** subroutine, **unlink** subroutine.

The File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

The Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* provides information on working with files.

The Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* explains working with directories and path names.

---

lint Command

Purpose

Checks C and C++ language programs for potential problems.

Syntax

lint [**-a**] [**-b**] [**-c**] [**-C**] [**-h**] [**-I***Key*] [**-n**] [**-o***Library*] [**-q***DBCS*] [**-p**] [**-t**] [**-u**] [**-v**] [**-w** *Class* [*Class* ...]] [**-x**] [**-MA**] [**-Nd***Number*] [**-Nl***Number*] [**-Nn***Number*] [**-Nt***Number*] [**-I***Directory*] [**-D***Name* [**=***Definition*]] [**-U***Name*] *File* ...

Description

The **lint** command checks C and C++ language source code for coding and syntax errors and for inefficient or non-portable code. You can use this program to:

- Identify source code and library incompatibility.
- Enforce type-checking rules more strictly than does the compiler.
- Identify potential problems with variables.
- Identify potential problems with functions.
- Identify problems with flow control.
- Identify legal constructions that may produce errors or be inefficient.
- Identify unused variable and function declarations.
- Identify possibly non-portable code.

> **Note:** Checking of C++ language files by the **lint** command requires the presence of the C Set++ Compiler package.

The inter-file usage of functions is checked to find functions that return values in some instances and not in others, functions called with varying numbers or types of arguments, and functions whose values are not used or whose values are used but not returned.

The **lint** command interprets file name extensions as follows:

- *File* names ending in **.c** are C language source files.
- *File* names ending in **.C** are C++ language source files.
- *File* names ending in **.ln** are non-ASCII files that the **lint** command produces when either the **-c** or the **-o** flag is used.

The **lint** command warns you about files with other suffixes and ignores them.

The **lint** command takes all the **.c**, **.C**, and **.ln** files and the libraries specified by **-l** flags and processes them in the order that they appear on the command line. By default, it adds the standard **llib-lc.ln** lint library to the end of the list of files. However, when you select the **-p** flag, the **lint** command uses the **llib-port.ln** portable library. By default, the second pass of **lint** checks this list of files for mutual compatibility; however, if you specify the **-c** flag, the **.ln** and **llib-l***x***.ln** files are ignored.

The **-c** and **-o** flags allow for incremental use of the **lint** command on a set of C and C++ language source files. Generally, use the **lint** command once for each source file with the **-c** flag. Each of these runs produces a **.ln** file that corresponds to the **.c** file and writes all messages concerning that source file. After you have run all source files separately through the **lint** command, run it once more, without the **-c** flag, listing all the **.ln** files with the needed **-l** flags. This writes all inter-file inconsistencies. This procedure works well with the **make** command, allowing it to run the **lint** command on only those source files modified since the last time that set of source files was checked.

The **lint** and **LINT** preprocessor symbols are defined to allow certain questionable code to be altered or removed for the **lint** command. Therefore, the **lint** and **LINT** symbols should be thought of as a reserved word for all code that is planned to be checked by **lint**.

The following comments in a C and C++ language source program change the way the **lint** command operates when checking the source program:

/*NOTREACHED*/ Suppresses comments about unreachable code.

/*VARARGS*Number***/**	Suppresses checking the following old style function declaration for varying numbers of arguments, but does check the data type of the first *Number* arguments. If you do not include a value for *Number*, the **lint** command checks no arguments (*Number*=0). The ANSI function prototypes should use the ellipsis to indicate unspecified parameters rather than this comment mechanism.
/*ARGSUSED*/	Suppresses warnings about function parameters not used within the function definition.
/*LINTLIBRARY*/	If you place this comment at the beginning of a file, the **lint** command does not identify unused functions and function parameters in the file. This is used when running the **lint** command on libraries.
/*NOTUSED*/	Suppresses warnings about unused external symbols, functions and function parameters in the file beginning at its point of occurrence. This is a superset of the **/*LINTLIBRARY*/** comment directive, but applies also to external symbols. It is useful for suppressing warnings about unused function prototypes and other external object declarations.
/*NOTDEFINED*/	Suppresses warnings about used, but undefined external symbols and functions in the file beginning at its point of occurrence.
/*LINTSTDLIB*/	Permits a standard prototype-checking library to be formed from header files by making function prototype declarations appear as function definitions. This directive implicitly activates both the **/*NOTUSED*/** and **/*LINTLIBRARY*/** comment directives to reduce warning noise levels.

The **lint** command warning messages give file name and line number. As each file goes through the first pass, warnings for each file and each line number are reported.

If you have not specified the **-c** flag, the **lint** command collects information gathered from all input files and checks it for consistency. At this point, if it is not clear whether a message stems from a given source file or from one of its included files, the **lint** command displays the source file name followed by a question mark.

ANSI programs that include many standard header files may wish to set the **-wD** flag to reduce the quantity of warnings about prototypes not used, and the **-n** flag to disable checking against the ANSI standard library. For non-ANSI programs, it is advisable to specify the **-wk** flag to reduce the amount of warnings concerning the absence of function prototypes.

Flags

-a	Suppresses messages about assignments of long values to variables that are not long.
-b	Suppresses messages about unreachable break statements.
-c	Causes the **lint** command to produce an **.ln** file for every **.c** file on the command line. These **.ln** files are the product of the first pass of the **lint** command only and are not checked for inter-function compatibility.
-C	Specifies to use the C++ libraries (in the **/usr/lpp//xlC/lib** directory).
-h	Does not try to detect bugs, improper style, or reduce waste.
-l*Key*	Includes the additional **llib-l***Key***.ln** lint library. You can include a lint version of the **llib-lm.ln** math library by specifying **-lm** on the command line or **llib-ldos.ln** library by specifying the **-ldos** flag on the command line. Use this flag to include local lint libraries when checking files that are part of a project having a large number of files. This flag does not prevent the **lint** command from using the **llib-lc.ln** library. The lint library must be in the **/usr/ccs/lib** directory.
-n	Suppresses the check for compatibility with either the standard or the portable lint libraries. This applies for both the ANSI and extended mode libraries.

-o*Library*
Causes the **lint** command to create the **llib-l***Library***.ln** lint library. The **-c** flag nullifies any use of the **-o** flag. The lint library produced is the input that is given to the second pass of the **lint** command. The **-o** flag simply causes this file to be saved in the named lint library. To produce a **llib-l***Library***.ln** without extraneous messages, use the **-x** flag. The **-v** flag is useful if the source files for the lint library are just external interfaces (for example, the way the **llib-lc** file is written). These flag settings are also available through the use of **lint** command comment lines.

-p
Checks for portability to other C language dialects.

-t
Checks for problematic assignments when porting from 32 to 64 bit. Only the following cases are checked:

- all shift / mask operations are flagged because some operations that work well in 32-bit may cause problems in 64-bit.
- warnings are given for the following type of assignments.

```
int = long
int = ptr
```

-u
Suppresses messages about functions and external variables that are either used and not defined or defined and not used. Use this flag to run the **lint** command on a subset of files of a larger program.

-v
Suppresses messages about function parameters that are not used.

-w*Class*[*Class...*]
Controls the reporting of warning classes. All warning classes are active by default, but can be individually deactivated by including the appropriate option as part of the *Class* argument. The individual options are listed as:

a Non-ANSI features.

c Comparisons with unsigned values.

d Declaration consistency.

h Heuristic complaints.

k Use for K+R type source code.

l Assignment of long values to variables that are not long.

n Null-effect code.

o Unknown order of evaluation.

p Various portability concerns.

r Return statement consistency.

s Storage capacity checks.

u Proper usage of variables and functions.

A Deactivate all warnings.

C Constants occurring in conditionals.

D External declarations are never used.

O Obsolescent features.

P Function prototype presence.

R Detection of unreachable code.

-x
Suppresses messages about variables that have external declarations but are never used.

-MA	Enforces the ANSI C language standard rules. The default mode is equal to the extended C mode. The ANSI mode prepends the standard ANSI library function prototypes in place of the default extended mode C lint library. The ANSI mode enforces a stricter inter-file object reference and provides definition linkage checks.
-Nd*Number*	Changes the dimension table size to *Number*. The default value of *Number* is 2000.
-Nl*Number*	Changes the number of type nodes to *Number*. The default value of *Number* is 8000.
-Nn*Number*	Increases the size of the symbol table to *Number*. The default value of *Number* is 1500.
-Nt*Number*	Changes the number of tree nodes to *Number*. The default value of *Number* is 1000.

In addition, the **lint** command recognizes the following flags of the **cpp** command (macro preprocessor):

-I*Directory*	Adds the *Directory* to the list of directories in which the **lint** command searches for the **#include** files.
-D*Name*[=*Definition*]	Defines the *Name*, as if by the **#define** file. The default of the *Definition* is the value of 1.
-q*DBCS*	Sets multibyte mode specified by the current locale.
-U*Name*	Removes any initial definition of the *Name*, where the *Name* is a reserved symbol that is predefined by the particular preprocessor.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To check a C program for errors, enter:
   ```
   lint command.c
   ```
2. To suppress some of the messages, enter:
   ```
   lint -v -x program.c
   ```
   This checks `program.c`, but does not display error messages about unused function parameters (**-v**) or unused externals (**-x**).
3. To check the program against an additional lint library, enter:
   ```
   lint -lsubs program.c
   ```
   This checks `program.c` against both the **/usr/ccs/lib/llib-lc.ln** standard lint library and **/usr/lib/llib-lsubs.ln** lint library.
4. To check against the portable library and an additional library, enter:
   ```
   lint -lsubs -p program.c
   ```
   This checks `program.c` against both the **/usr/ccs/lib/llib-port.ln** portable lint library and **/usr/lib/llib-lsubs.ln** lint library.
5. To check against a nonstandard library only, enter:
   ```
   lint -lsubs -n program.c
   ```
   This checks `program.c` against only **/usr/lib/llib-lsubs.ln**.

Files

/usr/lib/lint[12}	Programs
/usr/ccs/lib/llib-lansi	Declarations for standard ANSI functions (source)
/usr/ccs/lib/llib-lansi.ln	Declarations for standard ANSI functions (binary format)
/usr/ccs/lib/llib-lc	Declarations for standard functions (source)
/usr/ccs/lib/llib-lc.ln	Declarations for standard functions (binary format)
/usr/ccs/lib/llib-lcrses	Declarations for curses functions (source)
/usr/ccs/lib/llib-lcrses.ln	Declarations for curses functions (binary format)
/usr/ccs/lib/llib-lm	Declarations for standard math functions (source)
/usr/ccs/lib/llib-lm.ln	Declarations for standard math functions (binary format)
/usr/ccs/lib/llib-port	Declarations for portable functions (source)
/usr/ccs/lib/llib-port.ln	Declarations for portable functions (binary format)
/usr/lpp//xlC/lib	Directory containing C++ libraries
/var/tmp/*lint*	Temporary files

Related Information

The **cpp** command, **make** command.

---

listdgrp Command

Purpose

Displays devices of a device class.

Syntax

listdgrp *DeviceClass*

Description

Lists information about devices where the *DeviceClass* parameter referes to a object class of Customized Devices in the Device Configuration database.

Parameters

DeviceClass	Specifies the device class whose members will be displayed.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To list the devices in the **adapter** class, enter:

   ```
   listdgrp adapter
   ```

   The output looks similar to the following:

   ```
   a0
   sa1
   siokma0
   fda0
   scsi0
   ```

```
scsi1
bl0
sioka0
siota0
```

Files

/usr/bin/listdgrp	Contains the System V **listdgrp** command.

Related Information

The **chdev** command, **lsattr** command, **lsdev** command, **mkdev** command, **rmdev** command.

---

listvgbackup Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

listvgbackup [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **listvgbackup** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **listvgbackup** command also works for multi-volume backups such as multiple CDs, DVDs, or tapes.

The **listvgbackup -r** and **restorevgfiles** commands perform identical operations and should be considered interchangeable.

Flags

-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You may need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read will default to 100.
-B	Prints the volume group backup log to **stdout**. This flag will display the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command used to make backup, date, shrink size, full size of the backup, and recommended maintenance level (if any). **Note:** The shrink size is the size of the data on all filesystems. The full size is total size of each filesystem (unused + data).
-c	Produces colon-separated output. This flag only works with the **-l** and **-L** flags.
-d *path*	Specifies the directory path to which the files will be restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.
-D	Produces debug output.

-l Displays useful information about a volume group backup.

This flag requires the **-f** *device* flag. This flag causes **listvgbackup** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all filesystems. The full size is the total size of each filesystem (unused + data). The **-l** flag also displays the logical volume and filesystem information of the backed up volume group, equivalent to running ″**lsvg -l** *vgname*″.

-L Displays lpp fileset information about a **mksysb** backup only.

This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking ″**lslpp -l**″ on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.

-f *device* Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.

-r Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup will be restored. If the **-r** flag is not used, then executing the **listvgbackup** command only lists the files in the specified backup.

-s Specifies that the backup source is a user volume group and not rootvg.

-V Verifies a tape backup.

This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **listvgbackup** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

file_list Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory should be specified in the space-separated list. All files in the specified directory will be restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup located on the default device **/dev/rmt0**, enter:
   ```
   listvgbackup
   ```
2. To list the contents of the system backup located on device **/dev/cd1**, enter:
   ```
   listvgbackup -f /dev/cd1
   ```
3. To list the contents of the system backup located on device **/dev/cd1**, which is a user volume group that is not rootvg, enter:
   ```
   listvgbackup -f /dev/cd1 -s
   ```
4. To restore **/etc/filesystems** from the system backup located on device **/dev/cd1**, enter:
   ```
   listvgbackup -f /dev/cd1 -r ./etc/filesystems
   ```
5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is located on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter:
   ```
   listvgbackup -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```
6. To display colon separated lpp information about a **mksysb** backup tape located on **/dev/rmt0**, enter the following:
   ```
   lsmksysb -Lc -f /dev/rmt0
   ```
7. To display the volume group backup log to **stdout**, enter:
   ```
   lssavevg -B
   ```
8. To list volume group and general backup data about a backup located at **/tmp/mybackup**, enter:
   ```
   listvgbackup -l -f /tmp/mybackup
   ```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter:
   ```
   lsmksysb -V -f /dev/rmt0
   ```

Files

/usr/bin/listvgbackup	Contains the **listvgbackup** command

Related Information

The **restorevgfiles** command.

---

listX11input Command

Purpose

Lists X11 input extension records entered into the Object Data Manager (ODM) database.

Syntax

listX11input

Description

The **listX11input** command lists all X11 input extension records entered in the ODM database.

Error Codes

ODM could not open class	The ODM database is not stored in the **/usr/lib/objrepos** directory.

Related Information

The **addX11input** command, **deleteX11input** command.

---

llbd Daemon

Purpose

Manages the information in the local location broker database.

Syntax

llbd [**-family** *FamilyName*] [**-version**]

Description

The **llbd** daemon is part of the Network Computing System (NCS). It manages the local location broker (LLB) database, which stores information about NCS-based server programs running on the local host.

A host must run the **llbd** daemon to support the location broker forwarding function or to allow remote access (for example, by the **lb_admin** tool) to the LLB database. In general, any host that runs an NCS-based server program should run an **llbd** daemon, and **llbd** should be running before any such servers are started. Additionally, any network or internet supporting NCS activity should have at least one host running a global location broker daemon (**glbd**).

The **llbd** daemon is started in one of two ways:

- Through the System Resource Controller (the recommended method), by entering on the command line:

```
startsrc -s llbd
```

- By a person with root user authority entering on the command line:

```
/etc/ncs/llbd &
```

TCP/IP must be configured and running on your system before you start the **llbd** daemon. (You should start the **llbd** daemon before starting the **glbd** or **nrglbd** daemon.)

Flags

-listen *FamilyList*
: Restricts the address families on which an LLB listens. Use it only if you are creating a special configuration where access to an LLB is restricted to a subset of hosts in the network or internet.

  The *FamilyList* is a list of the address families on which the LLB will listen. Names in this list are separated by spaces. Possible family names include **ip**.

  If **llbd** is started without the **-listen** option, the LLB will listen on all address families that are supported both by NCS and by the local host.

-version
: Displays the version of NCS that this llbd belongs to, but does not start the daemon.

Files

/etc/rc.ncs Contains commands to start the NCS daemons.

Related Information

The **lb_admin** command, **startsrc** command..

The **glbd** (NCS) daemon, **nrglbd** (NCS) daemon.

---

ln Command

Purpose

Links files.

Syntax

To Link a File to a File

ln [**-f** | **-n**] [**-s**] *SourceFile* [*TargetFile*]

To Link a File or Files to a Directory

ln [**-f** | **-n**] [**-s**] *SourceFile* ... *TargetDirectory*

Description

The **ln** command links the file designated in the *SourceFile* parameter to the file designated by the *TargetFile* parameter or to the same file name in another directory specified by the *TargetDirectory* parameter. By default, the **ln** command creates hard links. To use the **ln** command to create symbolic links, designate the **-s** flag.

A symbolic link is an indirect pointer to a file; its directory entry contains the name of the file to which it is linked. Symbolic links may span file systems and may refer to directories.

If you are linking a file to a new name, you can list only one file. If you are linking to a directory, you can list more than one file.

The *TargetFile* parameter is optional. If you do not designate a target file, the **ln** command creates a new file in your current directory. The new file inherits the name of the file designated in the *SourceFile* parameter. See example 5.

Notes:

1. You cannot link files across file systems without using the **-s** flag.
2. If *TargetDirectory* is already a symbolic link to a directory, then the **ln** command treats the existing target as a file. This means that a command such as **ln -fs somepath/lname symdir** will not follow the existing symbolic link of **symdir**, instead it will create a new symbolic link from **somepath/lname** to **symdir**.

Flags

-f Causes the **ln** command to replace any destination paths that already exist. If a destination path already exists and the **-f** flag is not specified, the **ln** command writes a diagnostic message to standard error without creating a new link and continues to link the remaining *SourceFiles*.

-n Specifies that if the link is an existing file, do not overwrite the contents of the file. The **-f** flag overrides this flag. This is the default behaviour.

-s Causes the **ln** command to create symbolic links. A symbolic link contains the name of the file to which it is linked. The referenced file is used when an open operation is performed on the link. A **stat** call on a symbolic link returns the linked-to file; an **lstat** call must be done to obtain information about the link. The **readlink** call may be used to read the contents of a symbolic link. Symbolic links can span file systems and refer to directories.

> **Note:** Absolute path names must be used when specifying the *SourceFile* parameter for the **-s** flag. If the absolute path name is not designated, unexpected results may occur when the *SourceFile* and the *TargetFile* parameters are located in different directories. The source file does not need to exist before creating the symbolic link.

Exit Status

This command returns the following exit values:

0 All specified files were successfully linked.

>0 An error occurred.

Examples

1. To create another link (alias) to a file, enter:

   ```
   ln  -f chap1  intro
   ```

   This links `chap1` to the new name, `intro`. If `intro` does not already exist, the file name is created. If `intro` does exist, the file is replaced by a link to `chap1`. Then both the `chap1` and `intro` file names will refer to the same file. Any changes made to one also appear in the other. If one file name is deleted with the **rm** command, the file is not completely deleted since it remains under the other name.

2. To link a file to the same name in another directory, enter:

   ```
   ln  index  manual
   ```

   This links `index` to the new name, `manual/index`.

   > **Note:** `intro` in example 1 is the name of a file; `manual` in example 2 is a directory that already exists.

3. To link several files to names in another directory, enter:
   ```
   ln  chap2  jim/chap3  /home/manual
   ```
   This links `chap2` to the new name `/home/manual/chap2` and `jim/chap3` to `/home/manual/chap3`.
4. To use the **ln** command with pattern-matching characters, enter:
   ```
   ln  manual/* .
   ```
   This links all files in the `manual` directory into the current directory, **.** (dot), giving them the same names they have in the `manual` directory.

   Note: You must type a space between the asterisk and the period.
5. To create a symbolic link, enter:
   ```
   ln -s /tmp/toc toc
   ```
   This creates the symbolic link, `toc`, in the current directory. The `toc` file points to the `/tmp/toc` file. If the `/tmp/toc` file exists, the **cat** `toc` command lists its contents.

   To achieve identical results without designating the *TargetFile* parameter, enter:
   ```
   ln -s /tmp/toc
   ```

Files

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

Related Information

The **cp** command, **mv** command, **rm** command.

The **link** subroutine, **readlink** subroutine, **stat** subroutine, **symlink** subroutine.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes the structure and characteristics of directories in the file system.

Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes files, file types, and how to name files.

Linking Files and Directories in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* explains the concept of file linking.

Linking for Programmers in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs* discusses links from a programming viewpoint.

locale Command

Purpose

Writes information to standard output about either the current locale or all public locales.

Syntax

locale [**-O 64**][**-a** | **-m**] | [[**-c**] [**-k**] *Name* ...]

Description

The **locale** command writes information to standard output about either the current locale or all public locales. A public locale is a locale available to any application.

To write the name and value of each current locale category, do not specify any flags or variables. To write the names of all available public locales, specify the **-a** flag. To write a list of the names of all available character-mapping (charmap) files, specify the **-m** flag. These charmap filenames are suitable values for the **-f** flag specified with the **localedef** command.

To write information about specified locale categories and keywords in the current locale, specify the *Name* parameter. The *Name* parameter can be one of the following:

- A locale category, such as **LC_CTYPE** or **LC_MESSAGES**
- A keyword, such as **yesexpr** or **decimal_point**
- The **charmap** reserved word to determine the current character mapping

You can specify more than one *Name* parameter with the **locale** command.

If you specify the **locale** command with a locale category name and no flags, the **locale** command writes the values of all keywords in the locale category specified by the *Name* parameter. If you specify the **locale** command with a locale keyword and no flags, the **locale** command writes the value of the keyword specified by the *Name* parameter.

If the *Name* parameter is a locale category name or keyword, the **-c** and **-k** flags can determine the information displayed by the **locale** command.

Flags

-a Writes the names of all available public locales.

-c Writes the names of selected locale categories. If the *Name* parameter is a keyword, the **locale** command writes the name of the locale category that contains the specified keyword, and the value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the name of the specified locale category and the values of all keywords in the specified locale category.

-k Writes the names and values of selected keywords. If the *Name* parameter is a keyword, the **locale** command writes the name and value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the names and values of all keywords in the specified locale category.

-m Writes the names of all available character-mapping (charmap) files.

-ck Writes the name of the locale category, followed by the names and values of selected keywords. If the *Name* parameter is a keyword, the **locale** command writes the name of the locale category that contains the specified keyword, and the name and value of the specified keyword. If the *Name* parameter is a locale category, the **locale** command writes the name of the specified locale category and the names and values of all keywords in the specified locale category.

-O 64 Displays locale information as seen by a 64 bit executable. This should be identical to information as seen by a 32 bit executable.

Exit Status

This command returns the following exit values:

0 All the requested information was found and output successfully.

>0 An error occurred.

Examples

1. To retrieve the names and values of all the current locale environment variables, enter:

```
locale
```

If `locale_x` and `locale_y` are valid locales on the system, as determined with `locale -a`, and if the locale environment variables are set as follows:

```
LANG=locale_x
LC_COLLATE=locale_y
```

The **locale** command produces the following output:

```
LANG=locale_x
LC_CTYPE="locale_x"
LC_COLLATE=locale_y
LC_TIME="locale_x"
LC_NUMERIC="locale_x"
LC_MONETARY="locale_x"
LC_MESSAGES="locale_x"
LC_ALL=
```

> **Note:** When setting the locale variables, some values imply values for other locale variables. For example, if the **LC_ALL** locale variable is set to the **En_US** locale, all locale environment variables are set to the **En_US** locale. In addition, implicit values are enclosed in double quotes (″). Explicitly set values are not enclosed in double quotes (″). See ″Understanding Locale Environment Variables″ in *AIX 5L Version 5.2 National Language Support Guide and Reference* for more information.

2. To determine the current character mapping, enter:

   ```
   locale charmap
   ```

   If the **LC_ALL** locale variable is set to the C locale, the **locale** command produces the following output:

   ```
   ISO8859-1
   ```

3. To retrieve the value of the `decimal_point` delimiter for the current locale, enter:

   ```
   locale  -ck  decimal_point
   ```

   If the **LC_ALL** locale variable is set to the C locale, the **locale** command produces the following output:

   ```
   LC_NUMERIC
   decimal_point="."
   ```

Related Information

The **localedef** command.

Character Set Description (charmap) Source File Format and Locale Definition Source File Format in *AIX 5L Version 5.2 Files Reference*.

For specific information about the locale definition source file format, categories, and their locale variables, see the **LC_COLLATE** category, **LC_CTYPE** category, **LC_MESSAGES** category, **LC_MONETARY** category, **LC_NUMERIC** category, and **LC_TIME** category in *AIX 5L Version 5.2 Files Reference*.

National Language Support Overview, and Understanding Locale Environment Variables in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

---

localedef Command

Purpose

Converts locale and character set description (charmap) source files to produce a locale database.

Syntax

localedef [**-c**] [**-f** *Charmap*] [**-i** *SourceFile*] [**-L** *LinkOptions*] [**-m** *MethodFile*] *LocaleName*

Description

The **localedef** command converts source files that contain definitions of locale-dependent information (such as collation, date and time formats, and character properties) into a locale object file used at run-time. The locale object file created by the **localedef** command is then used by commands and subroutines that set the locale with the **setlocale** subroutine.

The **-i** *SourceFile* flag and variable specify the file that contains the source category definitions. If the **-i** flag is not specified, the file is read from standard input.

The **-f** *CharMap* flag and variable specify a file that maps character symbols to actual character encodings. Using the **-f** flag allows one locale source definition to be applicable to more than one code set. If the **-f** flag is not specified, the default value for the *CharMap* variable is ISO8859-1.

The *LocaleName* parameter specifies the locale name for the locale database generated by the **localedef** command from the specified source files. The *LocaleName* parameter can be either an absolute path name for the file location or a relative path name.

If a locale category source definition contains a copy statement and the statement names an existing locale installed in the system, the **localedef** command proceeds as though the source definition contained the valid category source definition for the named locale.

Notes:

1. The **localedef** command uses the C compiler to generate the locale database. Therefore, to use this command you must have the C compiler installed.
2. When replacing systemwide databases, it is advisable to do a soft reboot to ensure that the new locale is used throughout the system.

If an error is detected, no permanent output is created.

If warnings occur, permanent output is created when the **-c** flag is specified. The following conditions cause warning messages to be issued:

- A symbolic name not found in the file pointed to by the *Charmap* variable is used for the descriptions of the **LC_TYPE** or **LC_COLLATE** categories. This is an error condition for other categories.
- The number of operands to the **order_start** keyword exceeds the **COLL_WEIGHTS_MAX** limit.
- Optional keywords not supported by the implementation are present in the source file.

Flags

-c	Forces the creation of locale tables even if warning messages have been issued.
-f *CharMap*	Specifies the name of a file containing a mapping of character symbols and collating element symbols to actual character encodings. A locale is associated with one and only one code set. If this flag is not specified, the ISO 8859-1 code set is assumed. **Note:** The use of certain system-provided *CharMap* files is fully supported. However, while correctly defined user-provided *CharMap* files may work properly, the result of such use is not guaranteed.
-i *SourceFile*	Specifies the path name of a file containing the locale category source definitions. If this flag is not present, source definitions are read from standard input.
-L *LinkOptions*	Passes the specified link options to the **ld** command used to build the locale.

-m *MethodFile* Specifies the name of a method file that describes the methods to override when constructing a locale. The method file specifies user-supplied subroutines that override existing definitions, as well as a path name for the library containing the specified subroutines. The **localedef** command reads the method file and uses entry points when constructing the locale objects. The code set methods specified are also used in parsing the file pointed to by the *CharMap* variable.
Note: To create a 64-bit locale, the method file must specify the path of the library as a single archive that has two shared objects, one 32-bit and the other 64-bit, that contain the specified subroutines. Specifying separate paths to the 32-bit and 64-bit shared objects causes the **localedef** command to fail due to incompatible XCOFF format.

LocaleName Specifies the name of the locale to be created. This is the name that can subsequently be used to access this locale information.

Exit Status

The **localedef** command returns the following exit values:

0 No errors occurred and the locales were successfully created.
1 Warnings occurred and the locales were successfully created.
2 The locale specification exceeded limits or the code set or sets used were not supported by the implementation, and no locale was created.
3 The capability to create new locales is not supported.
>3 Warnings or errors occurred and no locales were created.

Examples

1. To create a locale called `Austin` from standard input and disregard warnings, enter:
   ```
   localedef  -c  Austin
   ```
2. To create a locale called `Austin` with `Austin.src` as source input, enter:
   ```
   localedef  -i  Austin.src  Austin
   ```

Related Information

The **ld** command, **locale** command.

The **setlocale** subroutine.

Character Set Description (charmap) Source File Format, Locale Definition Source File Format and Method Source File Format in *AIX 5L Version 5.2 Files Reference*.

For specific information about the locale categories and their keywords, see the **LC_COLLATE** category, **LC_CTYPE** category, **LC_MESSAGES** category, **LC_MONETARY** category, **LC_NUMERIC** category, and **LC_TIME** category for the locale definition source file format in *AIX 5L Version 5.2 Files Reference*.

National Language Support Overview for System Management in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

---

lock Command

Purpose

Reserves a terminal.

Syntax

lock [**-***Timeout*]

Description

The **lock** command requests a password from the user, reads it, and requests the password a second time to verify it. In the interim, the command locks the terminal and does not relinquish it until the password is received the second time or one of the following occurs:

- The timeout interval is exceeded.
- The command is killed by a user with appropriate permission.

The timeout default value is 15 minutes, but this can be changed with the -*Timeout* flag.

Flags

-*Timeout* Indicates the timeout interval in minutes, as specified by the *Timeout* parameter. The default value is 15 minutes.

Examples

1. To reserve a terminal under password control, enter:

   ```
   lock
   ```

   You are prompted for the password twice so the system can verify it. If the password is not repeated within 15 minutes, the command times out.
2. To reserve a terminal under password control, with a timeout interval of 10 minutes, enter:

   ```
   lock -10
   ```

Files

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

Related Information

The **passwd** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

---

lockd Daemon

Purpose

Processes lock requests.

Syntax

/usr/sbin/rpc.lockd [**-a** *NumberOfThreads*][**-t** *TimeOut*] [**-g** *GracePeriod*]

Description

The **lockd** daemon processes lock requests that are either sent locally by the kernel or remotely by another lock daemon. The **lockd** daemon forwards lock requests for remote data to the server site lock daemon through the RPC package. The **lockd** daemon then asks the **statd** (status monitor) daemon for monitor service. The reply to the lock request is not sent to the kernel until both the **statd** daemon and the server site **lockd** daemon reply. The **statd** daemon should always be started before the **lockd** daemon.

If either the status monitor or the server site lock daemon is unavailable, the reply to a lock request for remote data is delayed until all daemons become available.

When a server recovers, it waits for a grace period for all client site **lockd** daemons to submit reclaim requests. The client site **lockd** daemons, on the other hand, are notified of the server recovery by the **statd** daemon. These daemons promptly resubmit previously granted lock requests. If a **lockd** daemon fails to secure a previously granted lock at the server site, the **lockd** daemon sends a SIGLOST signal to the process.

The **lockd** daemon is started and stopped with the following System Resource Controller (SRC) commands:

```
startsrc -s rpc.lockd
stopsrc -s rpc.lockd
```

To modify the arguments passed to the **lockd** daemon when it is started, use the following command:

```
chssys -s rpc.lockd Parameters...
```

The status monitor maintains information on the location of connections as well as the status in the **/etc/sm** directory, the **/etc/sm.bak** file, and the **/etc/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.

By default **rpc.lockd** establishes a dynamic socket port number for receiving requests. Entries may be added to the **/etc/services** file specifying the port that **rpc.lock** will listen for requests on. The service name is **lockd** and a unique port number should be specified. The following entries in **/etc/services** file would specify that port 16001 be used for both **tcp** and **udp**.

```
lockd 16001/tcp
lockd 16001/udp
```

Flags

-a *NumberOfThreads*
: Specifies the maximum number of **rpc.lockd** threads. This can be useful on an NFS file system because all the **lockf()** or **flock()** calls (and other file locking calls) must go through the **rpc.lockd** daemon. This can severely degrade system performance because the lock daemon might not be able to handle thousands of lock requests per second. When a large number of concurrent processes are forked from a single script, the **rpc.lockd** value increases to the default maximum of 33. After the maximum **rpc.lockd** value is reached, any other lock requests must wait for the lockd for processing. This wait causes a recurring cycle of TCP timeouts thus allowing only one individual forked process to complete each cycle. The solution to this problem is to increase the maximum allowed number of **rpc.lockd** threads. This can be done with the following command:

  ```
  startsrc -s rpc.lockd -a NumberOfThreads (511 max.)
  ```

  or in the ODM with:

  ```
  chssys -s rpc.lockd -a NumberOfThreads (511 max.)
  ```

-g *GracePeriod*
: Uses the *GracePeriod* variable to specify the amount of time, in seconds, that the **lockd** daemon should wait for reclaim requests for previously granted locks. The default value of the *GracePeriod* variable is 45 seconds.

-t *TimeOut*
: Uses the *TimeOut* variable to specify the interval between retransmitting lock requests to the remote server. The default value for the *TimeOut* variable is 15 seconds.

Examples

1. To specify a grace period, enter:

   ```
   /usr/sbin/rpc.lockd  -g 60
   ```

   In this example, the grace period is set for `60` seconds.
2. To specify the amount of time the **lockd** daemon should wait before retransmitting a lock request, enter:

   ```
   /usr/sbin/rpc.lockd  -t 30
   ```

   In this example, the retransmissions occur after `30` seconds.

Files

/etc/services	Contains **lockd** parameter information entries.

Related Information

The **chssys** command, **nfso** command, **no** command.

The **statd** daemon.

The **fcntl** subroutine, **lockf** subroutine, **signal** subroutine.

List of NFS Commands.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

AIX 5L Version 5.2 Performance Management Guide.

---

locktrace Command

Purpose

Controls kernel lock tracing.

Syntax

locktrace [**-r** *ClassName* | **-s** *ClassName* | **-S** | **-R** | **-l**]

Description

The **locktrace** command controls which kernel locks are being traced by the **trace** subsystem. The default is to **trace** none. If the machine has been rebooted after running the `bosboot -L` command, kernel lock tracing can be turned on or off for one or more individual lock classes, or for all lock classes. If `bosboot -L` was not run, lock tracing can only be turned on for all locks or none. The **trace** events collected in this case when locks are taken or missed (hook id 112), and released (hook id 113) do not have the lock class name available.

Flags

-r *classname*	Turn off lock tracing for all the kernel locks belonging to the specified class. This option always fails if `bosboot -L` was not run.
-s *classname*	Turn on lock tracing for all the kernel locks belonging to the specified class. This option always fails if `bosboot -L` has not been executed.

-R	Turn off all lock tracing.
-S	Turn on lock tracing for all locks regardless of their class membership.
-l	List kernel lock tracing current status.

Examples

1. To start tracing the SEM_LOCK_CLASS, type:

   ```
   locktrace -s SEM_LOCK_CLASS
   ```

2. To stop all lock tracing, type:

   ```
   locktrace -R
   ```

File

/usr/bin/locktrace	Contains the **locktrace** command.
/usr/include/sys/lockname.h	Contains the **lock** class names.

Related Information

The **bosboot** command, and the **trace** command.

---

elogevent, logevent Command

Purpose

Logs event information generated by the event response resource manager (ERRM) to a specified log file.

Syntax

elogevent [**-h**] *log_file*

logevent [**-h**] *log_file*

Description

The **elogevent** script always return messages in English. The language in which the messages of the **logevent** script are returned depend on the locale settings.

These scripts capture event information that is posted by the event response resource manager (ERRM) in environment variables that are generated by the ERRM when an event occurs. These scripts can be used as actions that are run by an event response resource. They can also be used as templates to create other user-defined actions. See the *RSCT Administration Guide* to understand how an event response resource runs an action command.

Event information is returned about the ERRM environment variables that are described in the *RSCT Administration Guide*, and also includes the following:

Local Time
: Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

These scripts use the **alog** command to write event information to and read event information from the specified *log_file*.

Flags

–h Writes the 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 A required *log_file* is not specified.

2 The *log_file* path is not valid.

Restrictions

- These scripts must be run on the node where the ERRM is running.
- The user who runs These scripts must have write permission for the *log_file* where the event information is logged.

Standard Output

When the **-h** flag is specified, the script's usage statement is written to standard output.

Examples

1. To log information, specify **/tmp/event.log** in the Web-based System Manager interface. ERRM runs this command:

   ```
   /usr/sbin/rsct/bin/logevent/tmp/event.log
   ```

   The **/tmp/event.log** file does not need to exist when the command is run.
2. To see the contents of the **/tmp/event.log** file, run this command:

   ```
   alog -f /tmp/event.log -o
   ```

   The following sample output shows a warning event for the **/var** file system (a file system resource):

   ```
   ============================================================================
   Event reported at Mon Mar 27 16:38:03 2002

   Condition Name:                         /var space used
   Severity:                               Warning
   Event Type:                             Event
   Expression:                             PercentTotUsed>90

   Resource Name:                          /var
   Resource Class Name:                    IBM.FileSystem
   Data Type:                              CT_UINT32
   Data Value:                             91
   ```

Location

/usr/sbin/rsct/bin/elogevent Contains the **elogevent** script

/usr/sbin/rsct/bin/logevent Contains the **logevent** script

Related Information

Commands: **alog**

Books: see the *RSCT Administration Guide* for information about the event response resource manager (ERRM) and about how to use ERRM environment variables

---

logform Command

Purpose

Initializes a logical volume for use as a Journaled File System (JFS) or Enhanced Journaled File System (JFS2) log.

Syntax

logform [**-V** *vfstype*] *LogName*

Description

Attention: Executing the **logform** command on any Journaled File System (JFS) or Enhanced Journaled File System (JFS2) log device can result in data loss and file system corruption for all JFS or JFS2 file systems logged by the log device.

The **logform** command initializes a logical volume for use as a JFS or JFS2 log device. The **logform** command destroys all log records on existing log devices, which may result in file system data loss.

The *Logname* parameter specifies the absolute path to the logical volume to be initialized (for example, `/dev/jfslog1`).

Note: The only intended use for the **logform** command is to initialize a JFS log logical volume as a JFS log device or to initialize a JFS2 log logical volume as a JFS2 log device. The SMIT interface for creating a JFS or JFS2 and the **crfs** command allow only one JFS or JFS2 log device per volume group.

Flag

-V *vfstype* [`jfslog` | `jfs2log`]	If specified, *vfstype* indicates what type of file system the log should be formatted for. If this option is not specified, then the type is obtained from the logical volume's type. If there is no type specified for the logical volume, then the default vfs type from **/etc/vfs** is used.

Examples

1. To create a JFS logging device on a newly created volume group, first create a logical volume of type `jfslog`:

   ```
   mklv -t jfslog -y jfslog1 newvg 1
   ```

   This command creates a `jfslog` logical volume named `jfslog1` in the volume group `newvg`. The size of the logical volume is 1 logical partition.
2. To format the `jfslog1` logical volume once it has been created, enter:

   ```
   logform /dev/jfslog1
   ```

   The `jfslog1` logical volume is now ready to be used as a JFS log device.

Files

/etc/filesystems	Lists the known file systems and defines their characteristics, including the log device.

Related Information

The **crfs** command, **mkfs** command, **mklv** command.

The File System in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

The Mounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains mounting files and directories, mount points, and automatic mounts.

The Understanding Journaled File System Size Limitations in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*

---

logger Command

Purpose

Makes entries in the system log.

Syntax

logger [**-f** *File*] [**-i**] [**-p** *Priority*] [**-t** *Tag*] [*Message*]

Description

The **logger** command provides an interface to the **syslog** subroutine, which writes entries to the system log. A *Message* variable can be specified on the command line, which is logged immediately, or a *File* variable is read and each line of the *File* variable is logged. If you specify no flags or variables, the **logger** command will wait for you to enter a message from standard input. The messages returned by the **LOG_KERN** facility cannot be logged by this command.

Flags

-f *File*	Logs the specified *File* variable. If the *Message* variable is specified, this flag is ignored.
-i	Logs the process ID of the logger process with each line.
-p *Priority*	Enters the message with the specified priority. The *Priority* parameter may be a number or a *facility.level* priority specifier.
-t *Tag*	Marks every line in the log with the specified *Tag* parameter.
Message	Indicates the message to log. If this variable is not specified, the **logger** command logs either standard input or the file specified with the **-f** *File* flag.

Examples

1. To log a message indicating a system reboot, enter:
   ```
   logger System rebooted
   ```
2. To log a message contained in the `/tmp/msg1` file, enter:
   ```
   logger -f /tmp/msg1
   ```
3. To log the daemon facility critical level messages, enter:
   ```
   logger -pdaemon.crit
   ```

Exit Status

This command returns the following exit values:

0 Successful completion.
>0 An error occurred.

Files

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

Related Information

The **syslogd** daemon.

The **syslog** subroutine.

---

login Command

Purpose

Initiates a user session.

Syntax

login [**-h** *HostName*] [**-p**] [**-f** *User*] [*User* [*Environment*]]

Description

The **login** command (part of the **tsm** command) initiates sessions on the system for the user specified by the *User* parameter. You can also specify environment variables to be added to the user's environment. These are strings of the form *Variable*=*Value*. The **login** command is not normally entered on the command line.

Notes:

1. The **PATH**, **IFS**, **HOME**, and **SHELL** environment variables may not be initialized from the command line.
2. The **login** command supports multibyte user names. It is recommended that the system administrator restrict the user names to characters within the portable character set to remove any ambiguity.
3. If the **/etc/nologin** file exists, the system prevents the user from logging in and displays the contents of the **/etc/nologin** file. The system does allow the root user to log in if this file exists. The **/etc/nologin** file is removed when you reboot the system.

The **login** command can handle Distributed Computing Environment (DCE) user names of up to 1024 characters. DCE user names are stored in the **LOGIN** environment variable. Because DCE user names do not conform to standard operating system requirements, the first 8 characters of the DCE user name are stored in all standard operating system files and environments.

The **login** command performs the following functions:

Checks accounts The **login** command validates the user's account, ensuring authentication, logins enabled properly, and correct capacity for the port used for the login.

Authenticates users	The **login** command verifies the user's identity by using the system defined authentication methods for each user. If a password has expired, the user must supply a new password. If secondary authentication methods are defined, these methods are invoked but need not be successful in order to log in to the system.
Establishes credentials	The **login** command establishes the initial credentials for the user from the user database. These credentials define the user's access rights and accountability on the system.
Initiates a session	The **login** command initializes the user environment from the user database, from the command line, and from the **/etc/environment** configuration file; changes the current directory to the user's home directory (normally); and runs the user's initial program.

These functions are performed in the order given; if one fails, the functions that follow are not performed.

When a user logs in successfully, the **login** command makes entries in the **/etc/utmp** file that tracks current user logins and the **/var/adm/wtmp** file that is used for accounting purposes. The **login** command also sets the **LOGIN** and **LOGNAME** environment variables.

Information pertaining to each unsuccessful login is recorded in the **/etc/security/failedlogin** file. The information stored is the same as that in the **/etc/utmp** file, except that unrecognizable user names are logged as `UNKNOWN_USER`. This ensures that a password accidentally entered as a user name, for example, is not allowed into the system unencrypted.

After a successful login, the **login** command displays the message of the day, the date and time of the last successful and unsuccessful login attempts for this account, and the total number of unsuccessful login attempts for this account since the last successful login. These messages are suppressed if there is a **.hushlogin** file in your home directory.

The **login** command also changes the ownership of the login port to the user. This includes any ports noted as synonyms in the **/etc/security/login.cfg** file.

In order to preserve the integrity of the system, only one session at a time is allowed to be logged in to a port. This means that the **login** command entered from the shell prompt cannot succeed, as both the original session and the new login session would be on the same port. However, the **exec login** command succeeds because a new shell replaces the current one. The **login** command is typically a built-in shell command, causing the shell to replace itself.

> **Note:** Unless your terminal displays only uppercase letters, your user name should not consist of uppercase letters exclusively.

To log in with multibyte user names, you must first open a Japanese window (aixterm) and initiate a new login from the Japanese window.

Flags

-f *User*	Identifies a user who has already been authenticated. If the real ID of the login process is root (0), then the user is not authenticated.
-h *HostName*	Identifies the login as a remote login and specifies with the *HostName* variable the name of the machine requesting the login. This form of the login is used only by the **telnetd** and **rlogind** daemons.
-p	Preserves the current terminal type by setting it the value of the **$TERM** environment variable instead of the type contained in the **CuAt/PdAt** object classes database.

Security

Access Control: This command sets the setuid permission to the root user, grants executable (**x**) permission to any user, and is in the Trusted Computing Base.

Examples

To log in to the system as user `jamesd`, enter the following at the login prompt:

```
login: jamesd
```

If a password is defined, the password prompt appears. Enter your password at this prompt.

Files

/usr/sbin/login	Contains the **login** command.
/etc/utmp	Contains accounting information.
/var/adm/wtmp	Contains accounting information.
/etc/motd	Contains the message of the day.
/etc/passwd	Contains passwords.
$HOME/.hushlogin	Suppresses login messages.
/etc/environment	Contains user environment configuration information.
/etc/security/login.cfg	Contains port synonyms.
/etc/security/lastlog	Contains information pertaining to the most recent successful and unsuccessful login attempts.
/etc/security/failedlogin	Contains information pertaining to each unsuccessful login.

Related Information

The **getty** command, **setgroups** command, **setsenv** command, **su** command **tsm** command.

The **utmp, wtmp, failedlogin** file format, **lastlog** file format.

The **authenticate** subroutine, **setuid** subroutine.

Suppressing Login Messages in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Login and Logout Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Shells Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes what shells are, the different types of shells, and how shells affect the way commands are interpreted.

---

logins Command

Purpose

Displays user and system login information.

Syntax

logins [**-a**] [**-m**] [**-o**] [**-p**] [**-s**] [**-t**] [**-u**] [**-x**] [**-g** *Groups*] [**-l** *Logins*]

Description

The **logins** command displays information about user and system logins. By default, the **logins** command prints the following:

- Userr ID
- primary group name

- primary group ID
- the **/etc/passwd** account field on user information.

The output is sorted by user ID, displaying system logins followed by user logins.

Depending on the options chosen, the following fields can also be displayed:
- user or system login
- user ID number
- multiple group names
- multiple group IDs
- home directory
- login shell
- four password aging parameters
-
- /etc/passwd account field value (user name or other information)
- primary group name
- primary group ID

Flags

-a In addition to the default output, the **-a** flag adds two password expiration fields to the display. These fields show how many days a password can remain unused before it automatically becomes inactive and the date that the password will expire.

-g *Groups* Displays all users belonging to group, sorted by user ID. Multiple groups can be specified as a comma separated list. *Groups* must specify valid group names on the system. Comma separate names when specifying more than one group.

-l *Logins* Displays the requested login. Multiple logins can be specified as a comma-separated list. *Logins* must specify valid user names on the system.

-m Displays multiple group membership information.

-o Formats output into one line of colon separated fields.

-p Displays users without passwords.

-s Displays all system logins.

-t Sorts output by user name instead of by user ID.

-u Displays all user logins.

-x Prints an extended set of information about each selected user. Information for each user is printed on a separate line containing the home directory, login shell, and password aging information. The extended information includes the following:
- The password status
- The date on which the password was last changed
- The number of days required between changes
- The number of days allowed before a change is needed
- The number of days that the user will receive a password expiration warning message before the password expires

The password status is displayed in an abbreviated form as `PS` for logins with password, `NP` for no password or `LK` for locked.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To list all the logins with no passwords, enter:

   ```
   logins -p
   ```

   The output looks similar to the following:

   ```
   pwdless      204     staff           1
   nopwd        208     staff           1
   ```

   The **-p** option ensures that only logins with no passwords are listed.
2. To list all the system logins sorted by alphabetical order enter:

   ```
   logins -st
   ```

   The output looks similar to the following:

   ```
   adm             4       adm             4
   bin             2       bin             2
   daemon          1       staff           1
   lp              11      lp              11
   lpd             9       nobody          -2
   root            0       system          0
   sys             3       sys             3
   uucp            5       uucp            5
   ```

   The **-t** option prints out the logins sorted alphabetically and not by uid.
3. To list the login details of users ″root’ and ″admin″, enter:

   ```
   logins -l root,adm
   ```

   The output looks similar to the following:

   ```
   root            0       system          0
   adm             4       adm             4
   ```
4. To list the password aging details of users ″root″ and ″admin″ enter:

   ```
   logins -xl root,adm
   ```

   The output looks similar to the following:

   ```
   root            0       system          0
                           /
                           /usr/bin/ksh
                           PS 021102 0 0 0
   adm             4       adm             4
                           /var/adm
                           /sbin/sh
                           PS 000000 0 0 0
   ```

   The **-x** option ensures that extended password information for these logins are retrieved and printed in the output.
5. To display the multiple group information of a particular user in a colon separated format enter:

   ```
   logins -mol root,adm
   ```

   The output looks similar to the following:

   ```
   root:0:system:0::bin:2:sys:3:security:7:cron:8:audit:10:lp:11
   adm:4:adm:4:
   ```

   The **-m** option is used here to retrieve the multiple group information of a particular login (user). The **-o** option ensures that the output is displayed in colon separated format.
6. To display the users of the ″staff″ and ″sys″ groups in a colon separated format, sorted by user name, enter:

```
logins -tsog staff,sys
```

The output looks similar to the following:

```
bin:2:bin:2:
daemon:1:staff:1:
invscout:200:staff:1:
root:0:system:0:
sys:3:sys:3:
```

Files

/usr/bin/logins	Contains the **logins** command.
/etc/passwd	Contains the password file.
/etc/group	Contains the group file.

Related Information

The **lsuser** command, the **lsgroup** command.

---

logname Command

Purpose

Displays login name.

Syntax

logname

Description

The **logname** command displays the login name of the current process. This is the name that the user logged in with and corresponds to the **LOGNAME** variable in the system-state environment. This variable is only set when the user logs into the system.

Security

Access Control: This program is installed as a normal user program in the Trusted Computing Base.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

To display your login name to standard output, enter:

```
logname
```

Files

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

Related Information

The **getty** command, the **login** command, the **setgroups** command, the **setsenv** command, the **su** command, the **tsm** command.

---

logout Command

Purpose

Stops all processes on a port.

Syntax

logout

Description

The **logout** command terminates all processes either with the same controlling terminal as the present process or with all processes which have this terminal open. Processes that are not children of the present process are terminated upon access to the terminal. The present process is also terminated. If the **login** command user and the **logout** command user do not match, the **logout** command permission is denied, and the command stops.

Examples

From a shell started by the **ksh** or **bsh** command, enter:

```
logout
```

Files

/usr/bin/logout	Contains the **logout** command.
/etc/utmp	Contains a record of logged-in users.

Related Information

The **bsh** command, **getty** command, **init** command, **ksh** command, **login** command, **setgroups** command, **shell** command, **su** command, **tsh** command **tsm** command.

The **setuid** subroutine.

Shells Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes what shells are, the different types of shells, and how shells affect the way commands are interpreted.

---

look Command

Purpose

Finds lines in a sorted file.

Syntax

look [**-d**] [**-f**] *String* [*File ...*]

Description

The **look** command searches sorted files specified by the *File* parameter and prints all lines that begin with the string specified by the *String* parameter. The **look** command uses a binary search, therefore files specified by the *File* parameter must be sorted in the C locale collating sequence.

The **-d** and **-f** flags affect comparisons as in the **sort** command. This means a file must be sorted using the **-f** flag in the **sort** command before using the look command with the **-f** flag.

If the *File* parameter is not specified, the **/usr/share/dict/words** file is assumed with the collating sequence specified by the **-df** flags. The sort is completed using the current collating sequence. This should match the collating sequence used to produce the dictionary file. The **look** command limits the length of a word search to 256 characters.

Flags

-d Specifies dictionary order. Only letters, digits, tabs, and spaces are considered in comparisons.
-f Compares uppercase and lowercase letters as equivalent values. Case is not considered in the sorting so that initial-capital and all-capital words are not grouped together at the beginning of the output.

Note: To use the **look -f** command, the input file must be sorted with the **sort -f** command.

Example

To search for all lines in the `sortfile` file that begin with the letter `a`, enter:

```
look  a  sortfile
```

File

/usr/share/dict/words	Contains the default dictionary.

Related Information

The **grep** command, **sort** command.

Input and Output Redirection Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

lookbib Command

Purpose

Finds references in a bibliography.

Syntax

lookbib [**-n**] [*Database* ...]

Description

The **lookbib** command uses an inverted index made by the **indxbib** command to find sets of bibliographic references. The **lookbib** command reads keywords typed after the > prompt on the terminal, and retrieves records containing all these keywords. If nothing matches, nothing is returned except another > prompt.

The **lookbib** command asks if you need instructions and prints some brief information if you type a user-defined affirmative answer.

The *Database* parameter specifies files that contain bibliographic references, indexes, or similar types of information. It is possible to search multiple databases as long as they have a common index made by the **indxbib** command. In that case, only the first database name given to the **indxbib** command is specified to the **lookbib** command.

If the **lookbib** command does not find the index files (the **.i[abc]** files), it looks for a reference file with the same name as the first database, but without the suffixes. It creates a file with a **.ig** suffix, suitable for use with the **fgrep** command. It then uses this **fgrep** command file to find references. Using the **.ig** file is simpler but slower than using the **.i[abc]** files, and does not allow the use of multiple reference files.

Flags

-n Turns off the prompt for instructions.

Files

Database.**ia**	Contains the entry file.
Database.**ib**	Contains the posting file.
Database.**ic**	Contains the tag file.
Database.**ig**	Contains the output file.

Related Information

The **addbib** command, **indxbib** command, **refer** command, **roffbib** command, **sortbib** command.

lorder Command

Purpose

Finds the best order for member files in an object library.

Syntax

lorder [**-X** {**32**|**64**|**32_64**}] *File ...*

Description

The **lorder** command reads one or more object or library archive files, looking for external references and writing a list of paired file names to standard output. The first pair of files contains references to identifiers that are defined in the second file.

If object files do not end with **.o**, the **lorder** command overlooks them and attributes their global symbols and references to some other file.

Flags

-X *mode* Specifies the type of object file **lorder** 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 **lorder** to process any 64-bit objects and ignore 32-bit objects. The **-X** flag overrides the **OBJECT_MODE** variable.

Files

/tmp/sym* Contains temporary files.

Related Information

The **ar** command, **ld** command, **tsort** command, **xargs** command.

The **ar** file.

Subroutines Overview in *AIX 5L Version 5.2 General Programming Concepts*.

---

lp Command

The **lp** command includes information for the AIX Print Subsystem **lp** and the System V Print Subsystem **lp**.

AIX Print Subsystem lp Command

Purpose

Sends requests to a line printer.

Syntax

lp [**-c**] [**-d***Queue*] [**-m**] [**-n***Number*] [**-o***Option*] [**-s**] [**-t***Title*] [**-w**] [*Files*]

Description

The **lp** command arranges for the files specified by the *Files* parameter and their associated information (called a request) to be printed by a line printer. If you do not specify a value for the *Files* parameter, the **lp** command accepts standard input. The file name **-** (dash) represents standard input and can be specified on the command line in addition to files. The **lp** command sends the requests in the order specified. If the job is submitted to a local print queue, the **lp** command displays the following to standard output:

```
Job number is: nnn
```

where `nnn` is the assigned job number. To suppress the job number use the **-s** flag.

You can also change the print options for a previously submitted print job, specified by *RequestID*. If the request has finished printing, the change is rejected. If the request is already printing, it is stopped and restarted from the beginning, unless the **-P** flag is specified.

Flags

-c Copies the files to be printed immediately when the **lp** command is run. The **lp** command copies files only when requested. No links are created. If you specify the **-c** flag, be careful not to remove any of the files before they are printed. If you do not specify the **-c** flag, changes made to the files after the request is made appear in the printed output.

-d*Queue* Specifies the print queue to which a job is sent.

-f *FormName* [**-d any**]
Prints the request on the form *FormName*. The LP print service ensures that the form is mounted on the printer. If *FormName* is requested with a printer destination that cannot support the form, the request is rejected. If *FormName* has not been defined for the system, or if the user is not allowed to use the form, the request is rejected. For more information, see **lpforms**. When the **-d any** flag is specified, the request is printed on any printer that has the requested form mounted and can handle all other needs of the print request.

-H *SpecialHandling*
Prints the request according to the *SpecialHandling* value. Acceptable *SpecialHandling* values are:

hold
Holds the print the request until notified. If printing has already begun, it is stopped. Other print requests go ahead of a held request until it is resumed.

resume
Resumes a held request. If it had been printing when held, it will be the next request printed, unless subsequently bumped by an **immediate** request. The **-i** flag followed by a *RequestID* must be used whenever this argument is specified.

immediate
Print the request next. If more than one request is assigned **immediate**, the most recent request is printed first. If another request is currently printing, it must be put on hold to allow this immediate request to print. Available only to LP administrators.

-L *LocaleName*
Specifies the *LocaleName* as the locale to use with this print request. By default, *LocaleName* is set to the value of **LC_CTYPE**. If **LC_CTYPE** is not set, *LocaleName* defaults to the C locale.

-m
Sends mail (see the **mail** command) after the files are printed. By default, no mail is sent upon normal completion of the print request.

-n*Number*
Prints the number of copies of printed output. The default number of copies is 1.

-o*Options*
Specify printer-dependent *Options*. Several such *Options* may be collected by specifying the **-o** keyletter more than once (that is, **-o option[1] -o option[2]** ... **-o option[n]**), or by specifying a list of options with one **-o** keyletter enclosed in double quotes and separated by spaces (that is, **-o** ″**option[1]** ***option[2] . . . option[n]***″).

nobanner
Do not print a banner page with this request. The administrator can disallow this option at any time.

nofilebreak
Do not insert a form feed between the files given, if submitting a job to print more than one file. This option is not supported by printers configured to use the PS (PostScript) interface.

length=*scaled-decimal-number*
Print this request with pages *scaled-decimal-number* long. A *scaled-decimal-number* is an optionally scaled decimal number that gives a size in lines, characters, inches, or centimeters, as appropriate. The scale is indicated by appending the letter **i** for inches, or the letter **c** for centimeters. For length or width settings, an unscaled number indicates lines or characters; for line pitch or character pitch settings, an unscaled number indicates lines per inch or characters per inch (the same as a number scaled with **i**). For example, `length=66` indicates a page length of 66 lines, `length=11i` indicates a page length of 11 inches, and `length=27.94c` indicates a page length of 27.94 centimeters. This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

width=*scaled-decimal-number*
Print this request with pages ***scaled-decimal-number*** wide. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**.) This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

lpi=*scaled-decimal-number*
Print this request with the line pitch set to *scaled-decimal-number*. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**.) This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

cpi=pica|elite|compressed
Print this request with the character pitch set to **pica** (representing 10 characters per inch), **elite** (representing 12 characters per inch), or **compressed** (representing as many characters per inch as a printer can handle). There is not a standard number of characters per inch for all printers; see the **terminfo** database for the default character pitch for your printer.flag and is not supported by the PS (PostScript).

stty=*stty-option-list*
A list of options valid for the **stty** command; enclose the list with single quotes if it contains blanks.

-P *PageList*
Prints the pages specified in *PageList*. This flag can be used only if there is a filter available to handle it; otherwise, the print request is rejected. The *PageList* may consist of ranges of numbers, single page numbers, or a combination of both. The pages are printed in ascending order.

-q *priority-level*
Assigns this request *PriorityLevel* in the printing queue. The values of *PriorityLevel* range from 0 (highest priority) to 39 (lowest priority). If a priority is not specified, the default for the print service is used, as assigned by the system administrator. A priority limit may be assigned to individual users by the system administrator.

-R
Removes file(s) after submitting the print request.

-r
See **-T** *content-type* [**-r**].

-s
Suppresses the automatic return of job numbers. The **lp** command reports the job number as the default, the **-s** flag overrides the default.

-S { *CharacterSet* | *PrintWheel* [**-d any**]
Prints this request using the specified *CharacterSet* or *PrintWheel*. If a form was requested and it requires a character set or print wheel other than the one specified with the **-S** flag,the request is rejected.

For printers that take print wheels, if the print wheel specified is not one listed by the administrator as acceptable for the printer specified in this request, the request is rejected unless the print wheel is already mounted on the printer.

For printers that use selectable or programmable character sets, if the *CharacterSet* specified is not one defined in the Terminfo database for the printer (see **terminfo**), or is not an alias defined by the administrator, the request is rejected.

When the **-d any** flag is used, the request is printed on any printer that has the print wheel mounted or any printer that can select the character set, and that can handle all other needs of the request.

-T *ContentType* [**-r**]
Prints the request on a printer that supports the specified *ContentType*. If no printer accepts this type directly, a filter is used to convert the content into an acceptable type. If the **-r** flag is specified, a filter is not used. If **-r** is specified but no printer accepts the *ContentType* directly, the request is rejected. If the *ContentType* is not acceptable to any printer, either directly or with a filter, the request is rejected.

In addition to ensuring that no filters is used, the **-r** flag forces the equivalent of the **-o 'stty=-opost'** flag.

-t*Title*
Specifies printing the title of the file on the banner page of the output.

-w
Writes a message on the print requesters terminal after the files are printed. If the requester is not logged in, the **mail** command sends the message. If the user is logged in on multiple windows or terminals, the message may not be sent to the LFT where the command was issued. The message is sent to the first terminal on which the **writesrv** daemon sees the user to be logged in.

> **Note:** If the **-w** flag is used in conjunction with the **-m** flag, the print requester will only receive mail and will not get a message on the terminal.

-y *ModeList*

Prints this request according to the printing modes listed in *ModeList*. The allowed values for *ModeList* are locally defined. This flag may be used only if there is a filter available to handle it; otherwise, the print request is rejected.

The following list describes the *ModeList*:

-y reverse
Reverse the order in which pages are printed. This filter option is not supported by the LP Print Service.

-y landscape
Change the orientation of a physical page from portrait to landscape.

-y x=*number***,y=***number*
Change the default position of a logical page on a physical page by moving the origin.

-y group=*number*
Group multiple logical pages on a single physical page.

-y magnify=*number*
Change the logical size of each page in a document.

-o length=*number*
Select the number of lines in each page of the document.

-P *number*
Select, by page numbers, a subset of a document to be printed.

-n *number*
Print multiple copies of a document.

Examples

1. To print the **/etc/motd** file on printer `lp0` attached to device `dlp0`, enter:
   ```
   lp /etc/motd
   ```
2. To print 30 copies of the **/etc/motd** file using a copy of the file, and to notify the user that the job is completed using mail, enter:
   ```
   lp  -c  -m  -n30  -dlp0:lpd0 /etc/motd
   ```
3. To print the **/etc/motd** file using backend flags **-f** and **-a**, with a job title of `blah`, enter:
   ```
   lp  -t"blah"  -o -f  -o -a /etc/motd
   ```
4. To queue the `MyFile` file and return the job number, enter:
   ```
   lp myfile
   ```
5. To queue the `MyFile` file and suppress the job number, enter:
   ```
   lp  -s myfile
   ```

Exit Status

This command returns the following exit values:

0 All input files processed successfully.
>0 No output device is available, or an error occurred.

Files

/usr/sbin/qdaemon	Contains the queuing daemon.
/var/spool/lpd/qdir/*	Contains the queue requests.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains temporary copies of enqueued files.
/etc/qconfig	Contains the queue configuration file.
/etc/qconfig.bin	Contains digested, binary version of the **/etc/qconfig** file.

Related Information

The **cancel** command, **enable** command, **lpr** command, **lpstat** command, **mail** command.

The **writesrv** daemon.

The **/etc/qconfig** file.

Starting a Print Job in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Printer Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printers, Print Jobs, and Queues Overview for Users in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Spooler Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

System V Print Subsystem lp Command

Purpose

Sends print requests

Syntax

lp [*print-options*] [*files*]

lp -i *request-ID print-options*

Description

The first form of the **lp** command arranges for the named *files* and associated information (collectively called a request) to be printed. If filenames are not specified on the command line, the standard input is assumed. The standard input may be specified along with named *files* on the command line by listing the filenames and specifying **-** for the standard input. The *files* will be printed in the order in which they appear on the command line.

The LP print service associates a unique *request-ID* with each request and displays it on the standard output. This *request-ID* can be used later when canceling or changing a request, or when determining its status. See the **cancel** command for details about canceling a request, and **lpstat** for information about checking the status of a print request.

The second form of **lp** is used to change the options for a request submitted previously. The print request identified by the *request-ID* is changed according to the *print-options* specified with this command. The *print-options* available are the same as those with the first form of the **lp** command. If the request has finished printing, the change is rejected. If the request is already printing, it will be stopped and restarted from the beginning (unless the **-P** flag has been given).

Sending a print request

The first form of the **lp** command is used to send a print request either to a particular printer or to any printer capable of meeting all requirements of the print request.

Flags must always precede filenames, but may be specified in any order.

Printers for which requests are not being accepted will not be considered when the destination is **any**. (Use the **lpstat -a** command to see which printers are accepting requests.) However, if a request is destined for a class of printers and the class itself is accepting requests, then all printers in the class will be considered, regardless of their acceptance status.

For printers that take mountable print wheels or font cartridges, if you do not specify a particular print wheel or font with the **-S** flag, whichever one happens to be mounted at the time your request is printed will be used. The **lpstat -p** *printer* **-l** command is used to see which print wheels are available on a particular printer. The **lpstat -S -l** command is used to see what print wheels are available and on which printers. Without the **-S** flag, the standard character set is used for printers that have selectable character sets.

If you experience problems with jobs that usually print but on occasion do not print, check the physical connections between the printer and your computer. If you are using an automatic data switch or an A/B switch, try removing it and see if the problem clears.

Flags

-c Make copies of the *files* to be printed immediately when **lp** is invoked. Normally *files* will not be copied, but will be linked whenever possible. If the **-c** flag is not specified, the user should be careful not to remove any of the *files* before the request has been printed in its entirety. It should also be noted that if the **-c** flag is not specified, any changes made to the named *files* after the request is made but before it is printed will be reflected in the printed output.

-d *dest*

Choose *dest* as the printer or class of printers that is to do the printing. If *dest* is a printer, then the request will be printed only on that specific printer. If *dest* is a class of printers, then the request will be printed on the first available printer that is a member of the class. If *dest* is **any**, then the request will be printed on any printer that can handle it. Under certain conditions (unavailability of printers, file space limitations, and so on) requests for specific destinations may not be accepted (see **lpstat**). By default, *dest* is taken from the environment variable **LPDEST**. If **LPDEST** is not set, then *dest* is taken from the environment variable **PRINTER**. If **PRINTER** is not set, a default destination (if one exists) for the computer system is used. If no system default is set and **-T** is used, *dest* will be selected on the basis of *content-type* specified with the **-T** flag [see the description of **-T**]. Destination names vary between systems (see **lpstat**).

-f *form-name* **[-d any]**

Print the request on the form *form-name*. The LP print service ensures that the form is mounted on the printer. If *form-name* is requested with a printer destination that cannot support the form, the request is rejected. If *form-name* has not been defined for the system, or if the user is not allowed to use the form, the request is rejected. (see **lpforms**). When the **-d any** flag is given, the request is printed on any printer that has the requested form mounted and can handle all other needs of the print request.

-H *special-handling*

Print the request according to the value of *special-handling*. Acceptable values for *special-handling* are defined below:

hold Do not print the request until notified. If printing has already begun, stop it. Other print requests will go ahead of a held request until it is resumed.

resume

Resume a held request. If it had been printing when held, it will be the next request printed, unless subsequently bumped by an **immediate** request. The **-i** flag (followed by a *request-ID*) must be used whenever this argument is specified.

immediate

(Available only to LP administrators) Print the request next. If more than one request is assigned **immediate**, the most recent request will be printed first. If another request is currently printing, it must be put on hold to allow this immediate request to print.

-L *locale-name*

Specify *locale-name* as the locale to use with this print request. By default, *locale-name* is set to the value of **LC_CTYPE**. If **LC_CTYPE** is not set, *locale-name* defaults to the C locale.

-m Send mail after the files have been printed. By default, mail is not sent upon normal completion of the print request.

-n *number*

Print *number* copies of the output. The default is one copy.

-o *options*

Specify printer-dependent *options*. Several such *options* may be collected by specifying the **-o** keyletter more than once (that is, **-o option[1] -o option[2]** ... **-o option[n]**), or by specifying a list of options with one **-o** keyletter enclosed in double quotes and separated by spaces (that is, **-o** *"***option[1]** *option[2]* **. . .** *option[n]"*).

nobanner

Do not print a banner page with this request. The administrator can disallow this option at any time.

nofilebreak

Do not insert a form feed between the files given, if submitting a job to print more than one file. This option is not supported by printers configured to use the PS (PostScript) interface.

length=*scaled-decimal-number*

Print this request with pages *scaled-decimal-number* long. A *scaled-decimal-number* is an optionally scaled decimal number that gives a size in lines, characters, inches, or centimeters, as appropriate. The scale is indicated by appending the letter **i** for inches, or the letter **c** for centimeters. For length or width settings, an unscaled number indicates lines or characters; for line pitch or character pitch settings, an unscaled number indicates lines per inch or characters per inch (the same as a number scaled with **i**). For example, **length=66** indicates a page length of 66 lines, **length=11i** indicates a page length of 11 inches, and **length=27.94c** indicates a page length of 27.94 centimeters. This option may not be used with the **-f** option and is not supported by the PS (PostScript).

width=*scaled-decimal-number*

Print this request with pages *scaled-decimal-number* wide. (See the explanation of *scaled-decimal-numbers* in the discussion of **length**, above.) This option may not be used with the **-f** option and is not supported by the PS (PostScript).

lpi=*scaled-decimal-number*

Print this request with the line pitch set to *scaled-decimal-number*. (See the explanation of

scaled-decimal-numbers in the discussion of **length**, above.) This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

cpi=pica|elite|compressed
Print this request with the character pitch set to **pica** (representing 10 characters per inch), **elite** (representing 12 characters per inch), or **compressed** (representing as many characters per inch as a printer can handle). There is not a standard number of characters per inch for all printers; see the **terminfo** database for the default character pitch for your printer. This option may not be used with the **-f** flag and is not supported by the PS (PostScript).

stty=*stty-option-list*
A list of options valid for the **stty** command; enclose the list with single quotes if it contains blanks.

-P *page-list*
Print the pages specified in *page-list*. This flag can be used only if there is a filter available to handle it; otherwise, the print request will be rejected. The *page-list* may consist of ranges of numbers, single page numbers, or a combination of both. The pages will be printed in ascending order.

-q *priority-level*
Assign this request *priority-level* in the printing queue. The values of *priority-level* range from 0 (highest priority) to 39 (lowest priority). If a priority is not specified, the default for the print service is used, as assigned by the system administrator. A priority limit may be assigned to individual users by the system administrator.

-R Remove file(s) after submitting the print request. Use this flag with caution.

-r See **-T** *content-type* [**-r**] below.

-s Suppress the ``request id is ...'' message.

-S *character-set* **[-d any]**

-S *print-wheel* **[-d any]**
Print this request using the specified *character-set* or *print-wheel*. If a form was requested and it requires a character set or print wheel other than the one specified with the **-S** flag, the request is rejected.

For printers that take print wheels: if the print wheel specified is not one listed by the administrator as acceptable for the printer specified in this request, the request is rejected unless the print wheel is already mounted on the printer.

For printers that use selectable or programmable character sets: if the *character-set* specified is not one defined in the Terminfo database for the printer (see **terminfo**), or is not an alias defined by the administrator, the request is rejected.

When the **-d any** flag is used, the request is printed on any printer that has the print wheel mounted or any printer that can select the character set, and that can handle all other needs of the request.

-t *title* Print *title* on the banner page of the output. The default is no title. Enclose *title* in quotes if it contains blanks.

-T *content-type* **[-r]**
Print the request on a printer that can support the specified *content-type*. If no printer accepts this type directly, a filter will be used to convert the content into an acceptable type. If the **-r** flag is specified, a filter will not be used. If **-r** is specified but no printer accepts the *content-type* directly, the request is rejected. If the *content-type* is not acceptable to any printer, either directly or with a filter, the request is rejected.

In addition to ensuring that no filters will be used, the **-r** flag will force the equivalent of the **-o 'stty=-opost'** flag.

-w Write a message on the user's terminal after the *files* have been printed. If the user is not logged in, or if the printer resides on a remote system, then mail will be sent instead. Be aware that messages may be sent to a window other than the one in which the command was originally entered.

-y *mode-list*
Print this request according to the printing modes listed in *mode-list*. The allowed values for *mode-list* are locally defined. This option may be used only if there is a filter available to handle it; otherwise, the print request will be rejected.

The following list describes the *mode-list* options:

-y reverse
Reverse the order in which pages are printed. This filter option is not supported by the LP Print Service.

-y landscape
Change the orientation of a physical page from portrait to landscape.

-y x=*number***,y=***number*
Change the default position of a logical page on a physical page by moving the origin.

-y group=*number*
Group multiple logical pages on a single physical page.

-y magnify=*number*
Change the logical size of each page in a document.

-o length=*number*
Select the number of lines in each page of the document.

-P *number*
Select, by page numbers, a subset of a document to be printed.

-n *number*
Print multiple copies of a document.

Files

/var/spool/lp/*

Related Information

The **lpstat** command, and **mail** command.

---

lp.cat, lp.set, lp.tell Command

Note: This is a System V Print Subsystem command.

Purpose

Default printer filter used by interface programs.

Syntax

/usr/lib/lp/bin/lp.cat [**-r**] [*Rate*]

/usr/lib/lp/bin/lp.set *H_pitch V_pitch Width Length Charset*

Description

The **lp.cat** command is the default printer filter called by the interface programs. **lp.cat** reads the file to be printed on its standard input and writes it to the device to be printed on.

lp.cat handles the following signals:

- normal termination (**SIGTERM**)
- serial line hangup (**SIGHUP** due to DCD, Data Carrier Detect, dropping)
- interrupts (**SIGINT** and **SIGQUIT**)
- early pipe termination (**SIGPIPE**)

The **lp.cat** command aborts a printing job if it has to wait too long on output. The default maximum time to wait is calculated as twice the output buffer size (2 * 1024 bytes) divided by the smallest of the values of the transmission rate, print rate, or the specified *Rate* (all rates are in characters per second, CPS). A new maximum delay time may be defined by the *Rate* parameter provided that this increases the delay time. If *Rate* is 0 then the delay allowed is effectively infinite.

When the **-r** flag is specified, **lp.cat** reports the effective throughput in characters per second. This statistic is printed on the standard error after the transmission of every 1024 bytes.

Interface programs may call the **lp.set** command to set the character pitch (*H_pitch*), line pitch (*V_pitch*), page *Width*, page *Length*, and character set (*Charset*) on the printer.

The default units are characters per inch for the character and line pitch, number of columns for width, and number of lines for length. Units may be stated explicitly by appending the values with **c** for centimeters, and **i** for inches.

If it cannot set a particular characteristic, **lp.set** exits with a value of 1 and prints a one letter character code followed by newline to the standard error. The character codes are shown in the following table:

Character code	Printer characteristic not set
H	character pitch
L	page length
S	character set
V	line pitch
W	page width

Interface scripts use **lp.tell** to forward descriptions of printer faults to the print service. **lp.tell** sends everything that it reads on its standard input to the print service. The print service forwards the message as an alert to the print administrator.

Flags

-r Specifies reports the effective throughput in characters per second for the **lp.cat** command. This statistic is printed on the standard error after the transmission of every 1024 bytes.

Exit Status

The following table shows the possible exit values for **lp.cat**:

0 Normal exit; no error

1	Standard input not defined
2	Standard output not defined
3	Printer type (TERM) not defined or not in terminfo database
4	Standard input and output are identical
5	Write failed; printer may be off-line
6	Excessive delay caused timeout; printer may be off-line
7	Hangup (SIGHUP) detected; loss of carrier
8	Termination (SIGINT or SIGQUIT), or pipe closed early (SIGPIPE)

lp.set returns 0 if successful; otherwise it returns 1 on error.

lp.tell returns:

0	Normal exit; no error
1	Cannot open message queue to the print service
90	Cannot get printer name or key from print service
91	Cannot send message to print service
92	Cannot receive acknowledgment from print service
93	Corrupted acknowledgment received from print service
94	Print service reports message corrupted in transmission

Files

/usr/lib/lp/bin/lp.cat	Full pathname of **lp.cat**
/usr/lib/lp/bin/lp.set	Full pathname of **lp.set**
/usr/lib/lp/bin/lp.tell	Full pathname of **lp.tell**
/etc/lp/model	Printer interface programs directory
/etc/lp/interfaces	Printer interface programs directory

References

The **lp** command.

The **signal** subroutine.

The **terminfo** file.

---

lpadmin Command

Note: This is a System V Print Subsystem command.

Purpose

Configures the LP print service.

Syntax

Adding or Changing the Configuration of a Local Printer

lpadmin -p *Printer* **-v** *Device* [**-D** *Comment*] [**-A** *AlertType*] [**-W** *Minutes*] [**-c** *Class*] [**-e** *Printer1*] [**-F** *FaultRecovery*] [**-f allow:***FormList* | **-f deny***FormList*] [**-h**] [**-l** *Content-Type-List*] [**-i** *Interface*] [**-l**] [**-M -f** *Form-Name* [**-o** *File-break*]] [**-M -S** *Print-Wheel*] [**-m** *Model*] [**-O** *Copy-Options*] [**-o**

Print-Options] [**-o nobanner** | **-o banner**] [**-r** *Class*] [**-S** *List*] [**-s** *Server-Name* [**!***ServerPrinterName*]] [**-T** *Printer-Type-List*] [**-u allow:***Login-Id-List* | **-u deny:***Login-Id-List*]]

Adding or Changing the Configuration of a Remote Printer

lpadmin -p *Printer* **-s** *ServerName* [**!***ServerPrinterName*] **-v** *Device* [**-D** *Comment*] [**-A** *AlertType*] [**-W** *Minutes*] [**-c** *Class*] [**-e** *Printer1*] [**-F** *FaultRecovery*] [**-f allow:***FormList* | **-f deny***FormList*] [**-h**] [**-l** *Content-Type-List*] [**-i** *Interface*] [**-l**] [**-M -f** *Form-Name* [**-o** *Filebreak*]] [**-M -S** *Print-Wheel*] [**-m** *Model*] [**-O** *CopyOptions*] [**-o** *PrintOptions*] [**-o nobanner** | **-o banner**] [**-r** *Class*] [**-S** *List*] [**-T** *PrinterTypeList*] [**-u allow:***LoginIdList* | **-u deny:***LoginIdList*]] [**-v** *Device*]

Removing a Printer Destination

lpadmin -x *Destination*

Setting or Changing the System Default Destination

lpadmin -d [*Destination*]

Setting an Alert for a Print Wheel

lpadmin -S *Print-Wheel* **-A** *AlertType* [**-W** *Minutes*] [**-Q** *Requests*]

Description

The **lpadmin** command configures the LP print service by defining printers and devices. It is used to:

- Add and change printers
- Remove printers from the service
- Set or change the system default destination
- Define alerts for printer faults
- Mount print wheels
- Define printers for remote printing services

Printer and class names may be no longer than the maximum length filename allowed for the filesystem type you are using, and may consist of all printable characters except the space, slash, backslash, colon, semicolon, comma, asterisk, question mark, and tilde. The dash can be used in any position except the first position in a printer name.

For additional information about network printers, see the **lpsystem** command.

Adding or changing a printer

The **-p** *Printer* flag is used to configure a new printer or to change the configuration of an existing printer. When you use this form of the **lpadmin** command, you must select one of the following:

- **-v** *Device*, required to configure a local printer
- **-s** *ServerNname* [**!***ServerPrinterName*], required to configure a remote printer

Removing a printer destination

The **-x** *dest* flag removes the destination *dest* (a printer or a class), from the LP print service. If *dest* is a printer and is the only member of a class, then the class is deleted. If *dest* is **all**, all printers and classes are removed. No other parameters are allowed with **-x**.

Setting/changing the system default destination

The **-d** [*dest*] flag makes *dest*, an existing printer or class, the new system default destination. If *dest* is not supplied, then there is no system default destination. No other parameters are allowed with **-d**. To unset the system default printer, the user can enter the keyword **none**.

Setting an alert for a print wheel

The **-S** *Print-Wheel* flag is used with the **-A** *Alert-Type* flag to define an alert to mount the print wheel when there are jobs queued for it. If this command is not used to arrange alerting for a print wheel, no alert will be sent for the print wheel. See the other use of **-A** flag, with the **-p**.

The *Alert-Types* are the same as those available with the **-A** flag: **mail**, **write**, **quiet**, **none**, *shell-command*, and **list**. See the description of **-A**, for details about each.

The message sent appears as follows:

```
The print wheel Print-Wheel needs to be mounted
on the printer(s):
printer (integer1 requests)
integer2 print requests await this print wheel.
```

The printers listed are those that the administrator had earlier specified were candidates for this print wheel. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the printer list is the total number of requests awaiting the print wheel. It will be less than the sum of the other numbers if some requests can be handled by more than one printer.

If the *Print-Wheel* is **all**, the alerting defined in this command applies to all print wheels already defined to have an alert.

If the **-W** flag is not given, the default procedure is that only one message will be sent per need to mount the print wheel. Not specifying the **-W** flag is equivalent to specifying **-W once** or **-W 0**. If *Minutes* is a number greater than zero, an alert is sent at intervals specified by *minutes*.

If the **-Q** flag is also given, the alert is sent when a certain number (specified by the argument *requests*) of print requests that need the print wheel are waiting. If the **-Q** flag is not given, or *requests* is 1 or the word **any** (which are both the default), a message is sent as soon as anyone submits a print request for the print wheel when it is not mounted.

Flags

Notes:

- When creating a new printer, you must specify the **-v**, or **-s** flag. In addition, only one of the following may be supplied: **-e**, **-i**, or **-m**; if none of these three flags is supplied, the model standard is used.
- If **-s** and/or **-R** flags are specified, the following flags are not valid: **-A**, **-e**, **-F**, **-h**, **-i**, **-l**, **-M**, **-m**, **-o**, **-v**, and **-W**.

-A *AlertType* **[-W** *minutes*]

The **-A** flag defines an alert to inform the administrator when a printer fault is detected, and periodically thereafter, until the printer fault is cleared by the administrator. If an alert is not defined for a particular printer, mail is sent to user **lp** by default. The *AlertTypes* are:

mail Send the alert message via mail (see **mail**) to the administrator.

write Write the message to the terminal on which the administrator is logged in. If the administrator is logged in on several terminals, one is chosen arbitrarily.

quiet Do not send messages for the current condition. An administrator can use this option to temporarily stop receiving further messages about a known problem. Once the fault has been cleared and printing resumes, messages will again be sent when another fault occurs with the printer.

none Do not send messages; any existing alert definition for the printer is removed. No alert is sent when the printer faults until a different alert-type (except **quiet**) is used.

shell-command

Run the *shell-command* each time the alert needs to be sent. The shell command should expect the message in standard input. If there are blanks embedded in the command, enclose the command in quotes. The **mail** and **write** values for this option are equivalent to the values **mail** *login-ID* and **write** *login-ID* respectively, where *login-ID* is the current name for the administrator. This will be the login ID of the person submitting this command unless he or she has used the **su** command to change to another login ID. If the **su** command has been used to change the login ID, then the *login-ID* for the new login is used.

list Display the type of the alert for the printer fault. No change is made to the alert. The message sent appears as follows:

```
The printer Printer has stopped printing for the reason given
below.  Fix the problem and bring the printer back on line.
Printing has stopped, but will be restarted in a few minutes;
issue an enable command if you want to restart sooner.
Unless someone issues a change request

lp -i request-id -P . . .

to change the page list to print, the current request will be
reprinted from the beginning.

The reason(s) it stopped (multiple reasons indicate reprinted
attempts):

reason
```

The LP print service can detect printer faults only through an adequate fast filter and only when the standard interface program or a suitable customized interface program is used. Furthermore, the level of recovery after a fault depends on the capabilities of the filter.

If the *Printer* is **all**, the alerting defined in this command applies to all existing printers.

If the **-W** flag is not used to arrange fault alerting for *Printer*, the default procedure is to mail one message to the administrator of *Printer* per fault. This is equivalent to specifying **-W once** or **-W 0**. If *minutes* is a number greater than zero, an alert is sent at intervals specified by *minutes*.

-c *Class* Insert *Printer* into the specified *Class*. *Class* is created if it does not already exist.

-d [*Dest*] Makes *dest*, an existing printer or class, the new system default destination.

-D *Comment* Saves the *Comment* for display whenever a user asks for a full description of *Printer* (see **lpstat**). The LP print service does not interpret this comment.

-e *Printer1* Copies the interface program of an existing *Printer1* to be the interface program for *Printer*.

Note: Do not specify the **-i** and **-m** flags may not be specified with the **-e** flag.

-F *FaultRecovery* Specifies the recovery to be used for any print request that is stopped because of a printer fault, according to the value of *FaultRecovery*:

continue

Continue printing on the top of the page where printing stopped. This requires a filter to wait for the fault to clear before automatically continuing.

beginning

Start printing the request again from the beginning.

wait Disable printing on *Printer* and wait for the administrator or a user to enable printing again.

During the wait the administrator or the user who submitted the stopped print request can issue a change request that specifies where printing should resume. (See the **-i** flag of the **lp** command.) If no change request is made before printing is enabled, printing will resume at the top of the page where stopped, if the filter allows; otherwise, the request will be printed from the beginning.

The default value of *FaultRecovery* is **beginning**.

-f allow: *FormList* | **-f deny:***FormList*
Allows or denies the forms in *FormLIist* to be printed on *Printer*. By default no forms are allowed on a new printer.

For each printer, the LP print service keeps two lists of forms: an *allow-list* of forms that may be used with the printer, and a *deny-list* of forms that may not be used with the printer. With the **-f allow** flag, the forms listed are added to the allow-list and removed from the deny-list. With the **-f deny** flag, the forms listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the forms in the list may be used on the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the forms in the deny-list may not be used with the printer. All forms can be excluded from a printer by specifying **-f deny:all**. All forms can be used on a printer (provided the printer can handle all the characteristics of each form) by specifying **-f allow:all**.

The LP print service uses this information as a set of guidelines for determining where a form can be mounted. Administrators, however, are not restricted from mounting a form on any printer. If mounting a form on a particular printer is in disagreement with the information in the allow-list or deny-list, the administrator is warned but the mount is accepted. Nonetheless, if a user attempts to issue a print or change request for a form and printer combination that is in disagreement with the information, the request is accepted only if the form is currently mounted on the printer. If the form is later unmounted before the request can print, the request is canceled and the user is notified by mail.

If the administrator tries to specify a form as acceptable for use on a printer that doesn't have the capabilities needed by the form, the command is rejected.

The **lpadmin** command issues a warning when an invalid (nonexistent) form name is submitted with the **-f deny:** flag.

See the other use of **-f**, with the **-M** flag.

-h
Indicates that the device associated with the printer is hardwired. If neither of the mutually exclusive flags, **-h** and **-l**, is specified, this flag is assumed.

-I *Content-Type-List*
Allow *Printer* to handle print requests with the content types listed in a *Content-Type-List*. If the list includes names of more than one type, the names must be separated by commas or blank spaces. If they are separated by blank spaces, the entire list must be enclosed in double quotes.

The type **simple** is recognized as the default content type. A **simple** type of file is a data stream containing only printable ASCII characters and the following control characters.

Control Character	Octal Value	Meaning
backspace	10	move back one character, except at beginning of line
tab	11	move to next tab stop
linefeed (newline)	12	move to beginning of next line
form feed	14	move to beginning of next page
carriage return	15	move to beginning of current line

To prevent the print service from considering **simple** a valid type for the printer, specify either an explicit value (such as the printer type) in the *content-type-list*, or an empty list. If you do want **simple** included along with other types, you must include **simple** in the *content-type-list*.

Except for **simple**, each *content-type* name is freely determined by the administrator. If the printer type is specified by the **-T** option, then the printer type is implicitly considered to be also a valid content type.

-i *Interface*
Establish a new interface program for *Printer*. The *Interface* is the pathname of the new program. Do not specify the **-e** and **-m** flags with this flag.

-l
Indicates that the device associated with *Printer* is a login terminal. The LP scheduler (**lpsched**) disables all login terminals automatically each time it is started. The **-h** flag may not be specified with this flag.

-M -f *Form-Name* [**-a** [**-o filebreak**]]
Mounts the form *Form-Name* on *Printer*. Print requests that need the pre-printed form *Form-Name* are printed on *Printer*. If more than one printer has the form mounted and the user has specified **any** with the **-d** flag of the **lp** command as the printer destination, then the print request is printed on the one printer that also meets the other needs of the request.

The page length and width, and character and line pitches needed by the form are compared with those allowed for the printer, by checking the capabilities in the *terminfo* database for the type of printer. If the form requires attributes that are not available with the printer, the administrator is warned but the mount is accepted. If the form lists a print wheel as mandatory, but the print wheel mounted on the printer is different, the administrator is also warned but the mount is accepted.

If the **-a** flag is given, an alignment pattern is printed, preceded by the same initialization of the physical printer that precedes a normal print request. Printing is assumed to start at the top of the first page of the form. After the pattern is printed, the administrator can adjust the mounted form in the printer and press return for another alignment pattern (no initialization this time), and can continue printing as many alignment patterns as desired. The administrator can quit the printing of alignment patterns by typing `q`.

If the **-o filebreak** flag is given, a form feed is inserted between each copy of the alignment pattern. By default, the alignment pattern is assumed to correctly fill a form, so no form feed is added.

A form is unmounted either by mounting a new form in its place or by using the **-f none** flag. By default, a new printer has no form mounted.

See the other use of **-f** without the **-M**.

-M -S *Print-Wheel*
Mount the *Print-Wheel* on *Printer*. Print requests that need the *Print-Wheel* will be printed on *Printer*. If more than one printer has *Print-Wheel* mounted and the user has specified **any** with the **-d** flag of the **lp** command as the printer destination, then the print request is printed on the one printer that also meets the other needs of the request.

If the *Print-Wheel* is not listed as acceptable for the printer, the administrator is warned but the mount is accepted. If the printer does not take print wheels, the command is rejected.

A print wheel is unmounted either by mounting a new print wheel in its place or by using the **-S none** flag. By default, a new printer has no print wheel mounted.

See the other uses of the **-S** flag without the **-M**.

-m *Model*
Select *Model* interface program, provided with the LP print service, for the printer. DO not use the **-e** and **-i** flags with this flag. The following interface programs are available:

standard
generic printer interface

PS
interface for PostScript printers only

By default, the **standard** interface is used.

-O *Copy-Option*
The **-O** controls whether or not **lp** makes a copy of the user's file(s) when a print job is submitted. The *copy-option* can be either **copy** or **nocopy**. If **-O copy** is specified, the LP system always copies the user's source files to the spool area when a print job is submitted. If **-O nocopy** is specified, the files are copied only if the user specifies the **-c** flag of **lp** when submitting the job.

This flag sets the value of the **copy-files** parameter in the **/etc/default/lp** file. The value, which can be either **on** or **off**, is checked every time a print job is submitted.

-o *Printing-Option*
Specifies the, in the list below the default given to an interface program if the option is not taken from a preprinted form description or is not explicitly given by the user submitting a request (see **lp**). The only **-o** options that can have defaults defined are listed below.

length=*scaled-decimal-number*
width=*scaled-decimal-number*
cpi=*scaled-decimal-number*
lpi=*scaled-decimal-number*
stty=*'stty-option-list'*

scaled-decimal-number refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a trailing letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service:

- numbers that show sizes in centimeters, marked with a trailing **c**
- numbers that show sizes in inches, marked with a trailing **i**
- numbers that show sizes in units appropriate to use, without a trailing letter

that is, lines, characters, lines per inch, or characters per inch.

The first four default option values must agree with the capabilities of the type of physical printer, as defined in the *terminfo* database for the printer type. If they do not, the command is rejected.

The *stty-option-list* is not checked for allowed values, but is passed directly to the **stty** program by the standard interface program. Any error messages produced by **stty** when a request is processed (by the standard interface program) are mailed to the user submitting the request.

For each printing option not specified, the defaults for the following attributes are defined in the *terminfo* entry for the specified printer type.

```
length
width
cpi
lpi
```

The default for **stty** is

```
stty='9600 cs8 -cstopb -parenb ixon
-ixany opost -olcuc onlcr -ocrnl -onocr
-onlret -ofill nl0 cr0 tab0 bs0 vt0 ff0'
```

You can set any of the **-o** flags to the default values (which vary for different types of printers), by typing them without assigned values, as follows:

```
length=
width=
cpi=
lpi=
stty=
```

-o nobanner Allows a user to submit a print request specifying that no banner page be printed.

-o banner Forces a banner page to be printed with every print request, even when a user asks for no banner page. This is the default; you must specify **-o nobanner** if you want to allow users to be able to specify **-o nobanner** with the **lp** command.

-p *Printer* Configures a new printer changes the configuration of an existing printer.

-Q *Requests* Specifies that an alert be sent when a certain number of print *Requests* that need the print wheel are waiting.

-r *Class* Remove *Printer* from the specified *Class*. If *Printer* is the last member of *Class*, then *Class* is removed.

-S *List* Allows either the print wheels or aliases for character sets named in *List* to be used on the printer. The **-S** flag does not let you add items to a *List* specified with an earlier invocation of **-S**; instead, it replaces an existing *List* with a new one. Thus **-S** differs from the **-f**, **-u**, **allow**, and **deny** options, which allow you to modify existing lists of available forms and authorized users. Once you've run the **-S** flag, the print wheels and character sets specified, in *List*, on the current command line are the only ones available.

If the printer is a type that takes print wheels, then *List* is a comma or space separated list of print wheel names. Enclose the list with quotes if it contains blanks. These are the only print wheels considered mountable on the printer. You can always force a different print wheel to be mounted, however. Until the flag is used to specify a list, no print wheels is considered mountable on the printer, and print requests that ask for a particular print wheel with this printer is rejected.

If the printer is a type that has selectable character sets, then *List* is a comma or blank separated list of character set name mappings or aliases. Enclose the list with quotes if it contains blanks. Each mapping is of the form:

known-name=alias

The *known-name* is a character set number preceded by **cs**, such as `cs3` for character set three, or a character set name from the *Terminfo* database entry **csnm**. See **terminfo**. If this flag is not used to specify a list, only the names already known from the Terminfo database or numbers with a prefix of **cs** are acceptable for the printer.

If *List* is the word `none`, any existing print wheel lists or character set aliases is removed.

See the other uses of the **-S** with the **-M** flag.

-s *Server-Name* [**!***Server-Printer-Name*]
Specifies that you are configuring a remote printer. It makes a server printer accessible to users on your system. *Server-Name* is the name of the system on which the printer is located. It must be listed in the LP systems table. See **lpsystem**. *Server-Printer-Name* is the name used on the server system for that printer. For example, if you want to access *Printer1* on *Server1* and you want it called *Printer2* on your system, enter **-p** *Printer2* **-s** *Server1***!***Printer1*.

If *Server-Name* is a Netware server, defined as **-t nuc** using the **lpsystem** command, then *Server-Printer-Name* can be the name of a Netware queue or Netware printer.

-T *Printer-Type-List*
Identify the printer as being of one or more *Printer-Type*s. Each *Printer-Type* is used to extract data from the **terminfo** database; this information is used to initialize the printer before printing each user's request. Some filters may also use a *Printer-Type* to convert content for the printer. If this flag is not used, the default *Printer-Type* is `unknown`; no information will be extracted from **terminfo** so each user request is printed without first initializing the printer. Also, this flag must be used if the following are to work: **-o cpi**, **-o lpi**, **-o width**, and **-o length** flags of the **lpadmin** and **lp** commands, and the **-S** and **-f** flags of the **lpadmin** command.

If the *Printer-Type-List* contains more than one type, then the *content-type-list* of the **-I** option must either be specified as **simple**, as empty (**-I** ″″), or not specified at all.

-u allow:*Login-ID-List*
-u deny:*Login-ID-List*
Allows or denies the users in *Login-ID-List* access to the printer. By default all users on the local system are allowed on a new printer. The *Login-ID-List* parameter may include any or all of the following constructs:

login-ID
a user on the local system

*system-name***!***login-ID*
a user on system *system-name*

*system-name***!all**
all users on system *system-name*

all!*login-ID*
a user on all systems

all all users on the local system

all!all all users on all systems

For each printer the LP print service keeps two lists of users:

- An *allow-list* of people allowed to use the printer
- A *deny-list* of people denied access to the printer.

With the **-u allow** flag, the users listed are added to the allow-list and removed from the deny-list. With the **-u deny** flag, the users listed are added to the deny-list and removed from the allow-list.

If the allow-list is not empty, only the users in the list may use the printer, regardless of the contents of the deny-list. If the allow-list is empty, but the deny-list is not, the users in the deny-list may not use the printer. All users can be denied access to the printer by specifying **-u deny:all**. All users may use the printer by specifying **-u allow:all**.

-v *Device*
Specifies you are configuring a local printer. It associates a *Device* with *Printer*. *Device* is the pathname of a file that is writable by **lp**. The same *Device* can be associated with more than one printer.

-x *Dest*
Removes the destination *dest* (a printer or a class), from the LP print service.

Files

/var/spool/lp/*

/etc/lp

Related Information

The **accept** command, **enable** command, **lpsched**, command, **lpsystem** command.

---

lpc Command

Note: This is a System V Print Subsystem command.

Purpose

Provides (BSD) line printer control.

Syntax

/usr/ucb/lpc [*Command* [*Parameter* . . .]]

Description

The **lpc** command controls the operation of the printer or of multiple printers. The **lpc** command can be used to start or stop a printer, disable or enable a printer's spooling queue, rearrange the order of jobs in a queue, or display the status of each printer, along with its spooling queue and printer daemon.

With no parameters, the **lpc** command runs interactively, prompting with `lpc>`. If parameters are supplied, the **lpc** command interprets the first as a *Command* to execute; each subsequent parameter is taken as a *Parameter* for that command. The standard input can be redirected so that the **lpc** command reads *Command*s from a file.

*Command*s may be abbreviated to an unambiguous substring.

Note: The *printer* parameter is specified just by the name of the printer (as **lw**), not as you would specify it to **lpr** or **lpq** (not as **-Plw**).

? [*Command*. . .]

help [*Command* . . .]
Displays a short description of each command specified in the parameter list or, if no parameters are given, a list of the recognized commands.

abort [**all** | [*Printer* . . .]]
Terminates an active spooling daemon on the local host immediately and then disables printing (preventing new daemons from being started by **lpr**) for the specified printers. The **abort** command can only be used by a privileged user.

clean [**all** | [*Printer* . . .]]
Removes all files created in the spool directory by the daemon from the specified printer queues on the local machine. The **clean** command can only be used by a privileged user.

disable [**all** | [*Printer* . . .]]
Turns the specified printer queues off. This prevents new printer jobs from being entered into the queue by **lpr**. The **disable** command can only be used by a privileged user.

down [**all** | [*Printer*. . .]] [*Message*]
Turns the specified printer queue off, disables printing, and puts *Message* in the printer status file. The message does not need to be quoted. The remaining parameters are treated like **echo**. This is normally used to take a printer down and let others know why (**lpq** indicates that the printer is down, as does the **status** command).

enable [**all** | [*Printer*. . .]]
Enables spooling on the local queue for the listed printers so that **lpr** can put new jobs in the spool queue. The **enable** command can only be used by a privileged user.

exit
Exits from **lpc**.

quit
Quits from **lpc**.

restart [**all** | [*Printer*....]]	Attempts to start a new printer daemon. This is useful when some abnormal condition causes the daemon to die unexpectedly leaving jobs in the queue. This command can be run by any user.
start [**all** | [*Printer*...]]	Enables printing and starts a spooling daemon for the listed printers. The **start** command can only be used by a privileged user.
status [**all** | [*Printer*...].]	Displays the status of daemons and queues on the local machine. This command can be run by any user.
stop [**all** | [*Printer*...]]	Stops a spooling daemon after the current job completes and disable printing. The **stop** command can only be used by a privileged user.
topq *Printer* [*Job#*...] [*User*...]	Moves the print jobs specified by *Job#* or those jobs belonging to *User* to the top (head) of the printer queue. The **topq** command can only be used by a privileged user.
up [**all** | [*Printer*...]]	Enables everything and starts a new printer daemon. Undoes the effects of **down**.

Files

/var/spool/lp/*
/var/spool/lp/system/pstatus

Error Codes

?Ambiguous command	The abbreviation matches more than one command.
?Invalid command	A command or abbreviation is not recognized.
?Privileged command	The command can be executed only by the privileged user.
lpc: *printer***: unknown printer to the print service**	The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lptstat -p** to find the reason.
lpc: error on opening queue to spooler	The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lp/lpsched** is running.
lpc: Can't send message to LP print service	
lpc: Can't receive message from LP print service	Indicates that the LP print service has been stopped. Get help from the system administrator.
lpc: Received unexpected message from LP print service	It is likely there is an error in this software. Get help from system administrator.

Related Information

The **lpq** command, **lpr** command, **lprm** command, **lpsched** command.

---

lpd Command

Purpose

Provides the remote print server on a network.

Syntax

lpd [**-d**] [**-l**] [**-D** *DebugOutputFile*]

Description

The **lpd** daemon is the remote print server. It monitors port 515 for print requests. Each request is placed in a directory named **/var/spool/lpd**.

A computer on a network (host) that can create a Transmission Control Protocol/Internet Protocol (TCP/IP) data stream and use the **lpd** protocol can print remotely or act as a print server. As a security feature, the **lpd** daemon accepts print requests only from remote hosts that are listed in the local **/etc/hosts.equiv** or **/etc/hosts.lpd** file.

The **lpd** daemon can run on any host in the network; its function is to accept print requests from foreign hosts (on port 515). The **lpd** daemon handles each request by forking a child process. Remote requests are first checked against the **/etc/hosts.equiv** and **/etc/hosts.lpd** files for permission to print on the local host.

Changes can be made to the **/etc/hosts.equiv** and **/etc/hosts.lpd** files without restarting the system. To put changes to these files into effect without restarting the system, use the System Resource Controller (SRC) **refresh** command. This command causes the **/etc/hosts.equiv** and **/etc/hosts.lpd** database files to be reloaded and the changes implemented.

> **Note:** The queuing system does not support multibyte host names.

The **/etc/locks/lpd** file contains the process ID of the currently running instance of the **lpd** daemon. If the current machine becomes inoperable, you may need to remove the ID for the **lpd** daemon when the system starts up again. The error message displayed is `lpd: lock file or duplicate daemon`.

Manipulating the lpd Daemon with the System Resource Controller

The **lpd** daemon is a subsystem controlled by the System Resource Controller (SRC). The **lpd** daemon is a member of the TCP/IP system group.

Use the following SRC commands to manipulate the **lpd** daemon:

startsrc	Starts a subsystem, group of subsystems, or a subserver.
stopsrc	Stops a subsystem, group of subsystems, or a subserver.
refresh	Causes the subsystem or group of subsystems to reread the appropriate configuration file.
traceson	Enables tracing of a subsystem, group of subsystems, or a subserver.
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.

Flags

-d	Sends a status of Inactive to be logged with the SRC controller and sends error messages during socket communication setup failures to the user display.
-l	Sends a status of Active to be logged with the SRC controller and sends valid or invalid job request messages to the user display.
-D *DebugOutputFile*	Sends extensive debugging output used for problem determination to the file specified by *DebugOuputFile*. This should only be used during problem determination as the *DebugOuputFile* can grow large rapidly. If the output file specified already exists, new debugging output is appended to the end of it. If there are any problems creating or writing to the output file, the debugging option is ignored.

Examples

1. To start the **lpd** server daemon, enter:

   ```
   startsrc -s lpd
   ```

2. To start the **lpd** server daemon while enabling the display of certain error messages, enter:

   ```
   startsrc -s lpd -a " -d"
   ```

3. To send logging information to the **stderr** daemon, enter:

   ```
   startsrc -s lpd -a " -l"
   ```

4. To start the **lpd** server daemon in debugging mode with output going to **/tmp/dbglpd.out**, enter:

   ```
   startsrc -s lpd -a " -D /tmp/dbglpd.out"
   ```

Files

/usr/sbin/lpd	Specifies the path to the **lpd** daemon.
/dev/lp*	Contains the names of print devices.
/etc/hosts.equiv	Contains the names of hosts allowed to execute commands and print.
/etc/hosts.lpd	Contains the names of hosts allowed to print only.
/var/spool/lpd	Contains the spool directory for control, status, and data files.
/etc/locks/lpd	Contains the PID of the currently running **lpd** daemon. After a system crash, this PID may need to be deleted. The following error message indicates the problem: `lpd: lock file or duplicate daemon`

Related Information

Using the lpd Remote Subsystem in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Remote Printing Overview in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

---

lpfilter Command

Note: This is a System V Print Subsystem command.

Purpose

Administers filters used with the LP print service.

Syntax

lpfilter -f *FilterName* **-F** *PathName*

lpfilter -f *FilterName* **-**

lpfilter -f *FilterName* **-i**

lpfilter -f *FilterName* **-x**

lpfilter -f *FilterName* **-l**

Description

The **lpfilter** command is used to add, change, delete, and list a filter used with the LP print service. These filters are used to convert the content type of a file to a content type acceptable to a printer.

Flags

- (hyphen)	Adds or changes a filter as specified from standard input.
-f *FilterName*	Specifies the name of the filter to be added, changed, deleted, or listed.
-F *PathName*	Add or changes a filter as specified by the contents of the file pathname.
-i	Resets an original filter to its original settings.
-l	Lists a filter description.
-x	Deletes a filter.

The parameter **all** can be used instead of a *FilterName* with any of these flags. When **all** is specified with the **-F** or **-** flag, the requested change is made to all filters. Using **all** with the **-i** flag has the effect of restoring to their original settings all filters for which predefined settings were initially available. Using the **all** parameter with the **-x** flag results in all filters being deleted, and using it with the **-l** flag produces a list of all filters.

Adding or changing a filter

The filter named in the **-f** flag is added to the filter table. If the filter already exists, its description is changed to reflect the new information in the input.

The filter description is taken from the *PathName* if the **-F** flag is given or from the standard input if the **-** flag is specified. One of the two must be given to define or change a filter. If the filter named is one originally delivered with the LPprint service, the **-i** flag restores the original filter description.

When an existing filter is changed with the **-F** flag or the **-** flag, items that are not specified in the new information are left as they were. When a new filter is added with this command, unspecified items are given default values.

Filters are used to convert the content of a request into a data stream acceptable to a printer. For a given print request, the LP print service knows the following:

- Content in the request
- Name of the printer
- Type of the printer
- Types of content acceptable to the printer
- Modes of printing asked for by the originator of the request

It uses this information to find a filter or a pipeline of filters that converts the content into a type acceptable to the printer.

A list of items that provide input to this command and a description of each item follows. All lists are comma or space separated.

- Input types: *content-type-list*
- Output types: *content-type-list*
- Printer types: *printer-type-list*
- Printers: *printer-list*
- Filter type: *filter-type*
- Command: *shell-command*
- Flags: *template-list*

Input types	Gives the types of content that can be accepted by the filter. (The default is any.)
Output types	Gives the types of content that the filter can produce from any of the input content types. (The default is any.)

Printer types
Gives the type of printers for which the filter can be used. The LP print service restricts the use of the filter to these types of printers. (The default is any.)

Printers
Gives the names of the printers for which the filter can be used. The LP print service restricts the use of the filter to just the printers named. (The default is **any**.)

Filter type
Marks the filter as a **slow** filter or a **fast** filter. Slow filters are generally those that take a long time to convert their input. They are run unconnected to a printer to keep the printers from being tied up while the filter is running. If a listed printer is on a remote system, the filter type for it must have the value **slow**. Fast filters are generally those that convert their input quickly or those that must be connected to the printer when run. These are given to the interface program IP to run connected to the physical printer.

Command
Specifies the program to run to invoke the filter. The full program pathname as well as fixed flags must be included in the *shell-command*; additional flags are constructed, based on the characteristics of each print request and on the ″flags'' field. A command must be given for each filter.

The command must accept a data stream as standard input and produce the converted data stream on its standard output. This allows filter pipelines to be constructed to convert data not handled by a single filter.

Flags
Specifies the comma-separated list of templates used by the LP print service to construct flags to the filter from the characteristics of each print request listed in the table later.

In general, each template is of the following form:

keyword-pattern=replacement

The *keyword* names the characteristic that the template attempts to map into a filter-specific flag; each valid *keyword* is listed in the table below. A *pattern* is one of the following: a literal pattern of one of the forms listed in the table, a single asterisk (*), or a regular expression. If *pattern* matches the value of the characteristic, the template fits and is used to generate a filter-specific flag. The *replacement* is what is used as the flag.

Regular expressions are the same as those found in the **ed** or **vi** commands. This includes the \(. . . \) and *n* constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the ``&'', which can be used to copy the entire *pattern* into the *replacement*.

The *replacement* can also contain a ``*''. It too, is replaced with the entire *pattern*, just like the ``&'' of **ed** command.

lp flag	Characteristic	keyword	Possible patterns
-T	Content type (input)	INPUT	content-type
N/A	Content type (output)	OUTPUT	content-type
N/A	Printer type	TERM	printer-type
-d	Printer name	PRINTER	printer-name
-f, -o cpi=	Character pitch	CPI	integer

lp flag	Characteristic	keyword	Possible patterns
-f, -o lpi=	Line pitch	LPI	integer
-f, -o length=	Page length	LENGTH	integer
-f, -o width=	Page width	WIDTH	integer
-P	Pages to print	PAGES	page-list
-S	Character set Print wheel	CHARSET CHARSET	character-set-name print-wheel-name
-f	Form name	FORM	form-name
-y	Modes	MODES	mode
-n	Number of copies	COPIES	integer

For example, the template `MODES landscape = -l` shows that if a print request is submitted with the **-y landscape** flag, the filter is given the flag **-l**. As another example, the template `TERM * = -T *` shows that the filter is given the flag **-T** *printer-type* for whichever *printer-type* is associated with a print request using the filter.

As a last example, consider the template `MODES prwidth\=\(.*\) = -w\1`. Suppose a user gives the command **lp -y prwidth=10**

From the table above, the LP print service determines that the **-y** flag is handled by a **MODES** template. The **MODES** template here works because the pattern ``prwidth\=\(.*\)'' matches the **prwidth=10** given by the user. The *replacement* **-w\1** causes the LP print service to generate the filter flag **-w10**.

If necessary, the LP print service constructs a filter pipeline by concatenating several filters to handle the user's file and all the print flags. If the print service constructs a filter pipeline, the **INPUT** and **OUTPUT** values used for each filter in the pipeline are the types of the input and output for that filter, not for the entire pipeline.

Deleting a filter
The **-x** flag is used to delete the filter specified in *FilterName* from the LP filter table.

Listing a filter description
The **-l** flag is used to list the description of the filter named in *FilterName*. If the command is successful, the following message is sent to standard output:

```
Input types: content-type-list
   Output types: content-type-list
   Printer types: printer-type-list
   Printers: printer-list
   Filter type: filter-type
   Command: shell-command
   flags: template-list
```

If the command fails, an error message is sent to standard error.

Related Information
The **lpadmin** command.

---

lpforms Command

Note: This is a System V Print Subsystem command.

Purpose
Administer forms used with the LP print service.

Syntax
lpforms -f *FormName Options*

lpforms -f *FormName* **-A** *AlertType* [**-Q** *minutes*] [**-W** *requests*]

Description
The **lpforms** command is used to administer the use of preprinted forms, such as company letterhead paper, with the LP print service. A form is specified by its *FormName*. Users may specify a form when submitting a print request (see the **lp** command). The parameter **all** can be used instead of *FormName* with either of the command lines shown above. The first command line allows the administrator to add, change, and delete forms, to list the attributes of an existing form, and to allow and deny users access to particular forms. The second command line is used to establish the method by which the administrator is alerted that the form *FormName* must be mounted on a printer.

With the first **lpforms** command line, one of the following flags must be used:

Flags

- (hyphen)	Adds or changes form *FormName*, as specified by the information from standard input.
-F *pathname*	Adds or changes form *FormName*, as specified by the information in *pathname*.
-l	Lists the attributes of form *FormName*.
-x	Deletes form *FormName* (this flag must be used separately; it may not be used with any other flag).

Adding or changing a form
The **-F** *pathname* flag is used to add a new form, *FormName*, to the LP print service, or to change the attributes of an existing form. The form description is taken from *pathname* if the **-F** flag is given, or from the standard input if the **-** flag is used. One of these two flags must be used to define or change a form. *pathname* is the pathname of a file that contains all or any subset of the following information about the form:

```
Page length: scaled-decimal-number1
Page width: scaled-decimal-number2
Number of pages: integer
Line pitch: scaled-decimal-number3
Character pitch: scaled-decimal-number4
Character set choice: character-set/print-wheel [mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

The term ″scaled-decimal-number″ refers to a non-negative number used to indicate a unit of size. The type of unit is shown by a ″trailing″ letter attached to the number. Three types of scaled decimal numbers can be used with the LP print service: numbers that show sizes in centimeters (marked with a trailing ″c″); numbers that show sizes in inches (marked with a trailing ″i″); and numbers that show sizes in units appropriate to use (without a trailing letter), that is, lines, characters, lines per inch, or characters per inch.

Except for the last two lines, the above lines may appear in any order. The Comment and *comment* items must appear in consecutive order but may appear before the other items, and the ″Alignment pattern″ and the *content* items must appear in consecutive order at the end of the file. Also, the *comment* item may not

contain a line that begins with any of the key phrases above, unless the key phrase is preceded with a ″>″. Any leading ″>″ sign found in the *comment* are removed when the comment is displayed. Case distinctions in the key phrases are ignored.

When this command is issued, the form specified by *FormName* is added to the list of forms. If the form already exists, its description is changed to reflect the new information. Once added, a form is available for use in a print request, except where access to the form has been restricted, as described under the **-u** flag. A form may also be allowed to be used on certain printers only.

A description of each form attribute is below:

Page length and Page width
Before printing the content of a print request needing this form, the generic interface program provided with the LP print service initializes the physical printer to handle pages *scaled-decimal-number1* long, and *scaled-decimal-number2* wide using the printer type as a key into the *terminfo* database.

The page length and page width are also passed, if possible, to each filter used in a request needing this form.

Number of pages
Each time the alignment pattern is printed, the LP print service attempts to truncate the *content* to a single form by, if possible, passing to each filter the page subset of 1-*integer*.

Line pitch and Character pitch
Before printing the content of a print request needing this form, the interface programs provided with the LP print service initializes the physical printer to handle these pitches, using the printer type as a key into the *terminfo* database. Also, the pitches are passed, if possible, to each filter used in a request needing this form. *scaled-decimal-number3* is in lines per centimeter if a ″c″ is appended, and lines per inch otherwise; similarly, *scaled-decimal-number4* is in characters per centimeter if a ″c″ is appended, and characters per inch otherwise. The character pitch can also be given as **elite** (12 characters per inch), **pica** (10 characters per inch), or **compressed** (as many characters per inch as possible).

Character set choice
When the LP print service alerts an administrator to mount this form, it also mentions that the print wheel *print-wheel* should be used on those printers that take print wheels. If printing with this form is to be done on a printer that has selectable or loadable character sets instead of print wheels, the interface programs provided with the LP print service automatically selects or loads the correct character set. If **mandatory** is appended, a user is not allowed to select a different character set for use with the form; otherwise, the character set or print wheel named is a suggestion and a default only.

Ribbon color
When the LP print service alerts an administrator to mount this form, it also mentions that the color of the ribbon should be *ribbon-color*.

Comment
The LP print service displays the *comment* unaltered when a user asks about this form (see the **lpstat** command).

Alignment pattern

When mounting this form, an administrator can ask for the *content* to be printed repeatedly, as an aid in correctly positioning the preprinted form. The optional *content-type* defines the type of printer for which *content* had been generated. If *content-type* is not given, **simple** is assumed.

> **Note:** The content is stored as given and is readable only by the user *lp*.

When an existing form is changed with this command, items missing in the new information are left as they were. When a new form is added with this command, missing items gets the following defaults:

```
Page Length: 66
Page Width: 80
Number of Pages: 1
Line Pitch: 6
Character Pitch: 10
Character Set Choice: any
Ribbon Color: any
```

Deleting a form

The **-x** flag is used to delete the form *FormName* from the LP print service.

Listing form attributes

The **-l** flag is used to list the attributes of the existing form *FormName*. Because of the potentially sensitive nature of the alignment pattern, only the administrator can examine the form with this command. Other people may use the **lpstat** command to examine the non-sensitive part of the form description.

Allowing and denying access to a form

The **-u** flag, followed by the parameter **allow:***login-ID-list* or **-u deny:***login-ID-list* lets you determine which users are allowed to specify a particular form with a print request. This flag can be used with the **-F** or **-** flag.

The *login-ID-list* parameter may include any or all of the following constructs:

login-ID	A user on the local system
*system-name***!***login-ID*	A user on system *system-name*
*system-name***!all**	All users on system *system-name*
all!*login-ID*	A user on all systems
all	All users on the local system
all!all	All users on all systems

The default value of *login-ID-list* is **all**.

The LP print service keeps two lists of users for each form: an ″allow-list″ of people allowed to use the form, and a ″deny-list″ of people that may not use the form.

- If allow-list is present and *login-ID* is in it, access is allowed.
- If only deny-list is present and *login-ID* is not in it, access is allowed.
- If *login-ID* is in deny-list, access is denied.
- If neither allow-list or deny-list are present, access is denied.
- If both lists are present, and *login-ID* is in neither, access is denied.
- If only allow-list is present and *login-ID* is not in it, access is denied.

If the allow-list is not empty, only the users in the list are allowed access to the form, regardless of the contents of the deny-list. If the allow-list is empty but the deny-list is not, the users in the deny-list may not use the form (but all others may use it).

All users can be denied access to a form by specifying **-f deny:all**. All users can be allowed access to a form by specifying **-f allow:all**. (This is the default.)

Setting an alert to mount a form

The **-f** *FormName* flag is used with the **-A** *AlertType* flag to define an alert to mount the form when there are queued jobs which need it. If this flag is not used to arrange alerting for a form, no alert is sent for that form.

The method by which the alert is sent depends on the value of the *AlertType* parameter specified with the **-A** flag. The alert types are the same as those available with the **-A** flag to **lpadmin**: **mail**, **write**, **quiet**, **none**, *shell-command*, and **list**. See the description of **-A** on **lpadmin** for details about each.

The message sent appears as follows:

```
The form FormName needs to be mounted
on the printer(s):
printer (integer1 requests).
integer2 print requests await this form.
Use the ribbon-color ribbon.
Use the print-wheel print wheel, if appropriate.
```

The printers listed are those that the administrator had earlier specified were candidates for this form. The number *integer1* listed next to each printer is the number of requests eligible for the printer. The number *integer2* shown after the list of printers is the total number of requests awaiting the form. It is less than the sum of the other numbers if some requests can be handled by more than one printer. The *ribbon-color* and *print-wheel* are those specified in the form description. The last line in the message is always sent, even if none of the printers listed use print wheels, because the administrator may choose to mount the form on a printer that does use a print wheel.

Where any color ribbon or any print wheel can be used, the statements above read:

```
Use any ribbon.
   Use any print-wheel.
```

If *FormName* is **any**, the alerting defined in this command applies to any form for which an alert has not yet been defined. If *FormName* is **all**, the alerting defined in this command applies to all forms.

If the **-W** flag is not given, the default procedure is that only one message is sent per need to mount the form. Not specifying the **-W** flag is equivalent to specifying **-W once** or **-W 0**. If *minutes* is a number greater than 0, an alert is sent at intervals specified by *minutes*.

If the **-Q** flag is also given, the alert is sent when a certain number (specified by the parameter *requests*) of print requests that need the form are waiting. If the **-Q** flag is not given, or the value of *requests* is 1 or **any** (which are both the default), a message is sent as soon as anyone submits a print request for the form when it is not mounted.

Listing the current alert

The **-f** flag, followed by the **-A** flag and the parameter **list** is used to list the type of alert that has been defined for the specified form *FormName*. No change is made to the alert. If *FormName* is recognized by the LP print service, one of the following lines is sent to the standard output, depending on the type of alert for the form.

```
When requests requests are queued:
alert with shell-command every minutes minutes

When requests requests are queued:
write to user-name every minutes minutes
```

```
When requests requests are queued:
mail to user-name every minutes minutes

No alert
```

The phrase ″every *minutes* minutes″ is replaced with ″once″ if *minutes* (**-W** *minutes*) is 0.

Terminating an active alert

The **-A quiet** flag is used to stop messages for the current condition. An administrator can use this flag to temporarily stop receiving further messages about a known problem. Once the form has been mounted and then unmounted, messages are again sent when the number of print requests reaches the threshold *requests*.

Removing an alert definition

No messages are sent after the **-A none** flag is used until the **-A**flag is given again with a different *AlertType*. This can be used to permanently stop further messages from being sent as any existing alert definition for the form is removed.

Related Information

The **lpadmin** command.

The **terminfo** file.

---

lpmove Command

Note: This is a System V Print Subsystem command.

Purpose

Moves print requests.

Syntax

lpmove *Requests Destination*

lpmove *Destination1 Destination2*

Description

The **lpmove** command moves requests that were queued by **lp** between LP destinations. This command moves a specific *Request* to the specified *Destination*. *Requests* are request-IDs returned by **lp**. You can also attempt to move all requests for *Destination1* to *Destination2*. This form of the **lpmove** command causes **lp** to reject any new requests for *Destination1*.

Note: When moving requests, **lpmove** never checks the acceptance status (see **accept**) of the new destination. Also, the request-IDs of the moved requests are not changed, so you can still find their requests. The **lpmove** command does not move requests that have options (such as content type and form required) that cannot be handled by the new destination.

If a request was originally queued for a class or the special destination **any** and the first form of **lpmove** was used, the destination of the request is changed to *New-Ddestination*. A request thus affected is printable only on *New-Destination* and not on other members of the class or other acceptable printers if the original destination was **any**.

Files

/var/spool/lp/*

Related Information

The **accept** command, **enable** command, **lp** command, **lpadmin** command, **lpstat** command.

---

lppchk Command

Purpose

Verifies files of an installable software product.

Syntax

lppchk { **-c**[**u**] | **-f** | **-l** [**u**] | **-v** } [**-m** [**1** | **2** | **3**]] [**-O** { [**r**] [**s**] [**u**] }] [*ProductName* [*FileList* ...]]

Description

The **lppchk** command verifies that files for an installable software product (fileset) match the Software Vital Product Data (SWVPD) database information for file sizes, checksum values, or symbolic links. A fileset is a separately installable option of a software package.

Flags

-c Performs a checksum operation on the *FileList* items and verifies that the checksum and the file size are consistent with the SWVPD database.

-f Checks that the *FileList* items are present and the file size matches the SWVPD database.

-l Verifies symbolic links for files as specified in the SWVPD database.

-m [**1**|**2**|**3**] Displays three levels of information. The levels are as follows:

- **1** Error messages only (default).
- **2** Error messages and warnings.
- **3** Error messages, warnings and informational messages.

-O {[**r**][**s**][**u**]} Verifies the specified parts of the program. This flag is not needed with standalone systems because without this option all parts are verified by default. The flags specify the following parts:

- **r** Indicates the **/** (root) part is to be verified.
- **s** Indicates the **/usr/share** part is to be verified.
- **u** Indicates the **/usr** part is to be verified.

-u Updates the SWVPD with new checksum or size information from the system when the system information does not match the SWVPD database. This flag sets symbolic links that are found to be missing. This flag is valid with only the **-c** or **-l** flag.

-v Verifies that the **/** (root), **/usr** and **/usr/share** parts of the system are valid with each other. In other words, this flag verifies that all software products installed on the **/** (root) file system are also installed on the **/usr** file system and, conversely, all the software products installed in the **/usr** file system are also installed on the **/** (root) file system. You cannot specify *FileList* items with this flag. This flag also verifies requisites.
Note: Only one of the **-c**, **-f**, **-l**, and **-v** flags can be specified with each use of the **lppchk** command.

Parameters

FileList Specifies the file or files to check. This parameter is a list of file names separated by spaces. The file names can be a single name or a pair of names separated by a colon. The first form specifies a simple file and the second form specifies a member of an archive file, where the first name specifies the member and the second name specifies the archive file that contains the member. The full path name of the file or files must be specified. To specify multiple files you can use the pattern-matching characters ***** (asterisk) and **?** (question mark), but they should be enclosed in a pair of 's (single quotes). Single quotes are recommended to prevent the korn shell wildcard expansion.

If this parameter is omitted, all files of a software product are checked. If this parameter is specified, it must be preceded by a software product name.

ProductName Specifies the name of the software product whose files are to be checked. If this parameter is omitted, all software products in the SWVPD are checked. To specify multiple software products you can use the pattern-matching characters ***** (asterisk) and **?** (question mark), but they must be enclosed in a pair of 's (single quotes) to prevent the shell from expanding them.

Return Values

The **lppchk** command returns zero if no errors were found. Any other return value indicates an error was found.

Examples

1. To verify all files that comprise the **X11.fnt** package, type:

   ```
   lppchk -c X11.fnt
   ```

2. To verify the symbolic links of all software products whose names begin with **X11**, type:

   ```
   lppchk -l 'X11*'
   ```

3. To verify that all filesets have all required requisites and are completely installed, type:

   ```
   lppchk -v
   ```

Files

/etc/objrepos/lpp Specifies installation information of all software products on the root.

/usr/lib/objrepos/lpp Specifies installation information of all software products on the **/usr** file system.

/usr/share/lib/objrepos/lpp Specifies installation information of all software products on the **/usr/share** file system.

/etc/objrepos/product Specifies installation and update information of all software products on the root.

/usr/lib/objrepos/product Specifies installation and update information of all software products on the **/usr** file system.

/usr/share/lib/objrepos/product Specifies installation and update information of all the software products on the **/usr/share** file system.

/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.

lppmgr Command

Purpose

Manages an existing installp image source.

Syntax

lppmgr -d *DirectoryOrDevice* [**-r** | **-m** *MoveDirectory*] { [**-x**] [**-X**] [**-l**] [**-u**] [**-b**] [**-k** *LANG*] } [**-p**] [**-t**] [**-s**] [**-V**] [**-D**]

Description

lppmgr is designed to perform the following functions on an existing installp image source (also known as an **lpp_source** in the NIM environment):

1. Remove duplicate updates (**-u** Flag).
2. Remove duplicate base levels (**-b** Flag).
3. Eliminating updates which are the same level as bases of the same fileset. Such updates can create conflicts that lead to installation failure (**-u** Flag).
4. Remove message and locale filesets other than the language you specify (**-k** Flag).
5. Remove superseded filesets (**-x** Flag).
6. Remove non-system images from a NIM lpp_source resource (**-X** Flag).

By default, **lppmgr** will list all images filtered by the above routines. The ″**-r**″ flag can be used to remove the filtered images and the ″**-m**″ flag can be used to move the images to another location.

Note: **lppmgr** is not intended to replace **bffcreate**, install anything, or work with installed filesets. It is also not intended to address any issues other than those aforementioned. Before using the ″**-X**″ flag, you should have a good understanding of NIM, system images (known as SIMAGES in NIM), and the workings of a NIM **lpp_source** resource.

Flags

-b — Causees **lppmgr** to filter for base level duplicates.

-D — Specifies debug mode. This flag is for debugging the **lppmgr** script.
Note: Debug. This produces a large quantity of output and greatly reduces lppmgr performance. It is not useful for normal operations.

-d *DeviceOrDirectory* — Specifies the device or directory where the installp images reside. Currently this can be any directory, NFS mount point, or cdrom device. If the directory is not writable, you should use the **-t** flag. If the target of your operation is a NIM **lpp_source** resource, you should specify the **lpp_source** location (see the **lsnim** command). This flag is required for all operations.

-k *LANG* — Keeps only the message and locale images for the language specified by *LANG*. All other languages are filtered.

-l — Lists filtered images only. By default, **lppmgr** will only list all filtered image files unless the ″**-r**″ or ″**-m**″ flag is specified. The ″**-l**″ flag will override the ″**-r**″ or ″**-m**″ flag.

-m *Directory* — Moves filtered files to *Directory*. The location specified by *Directory* can be any writable directory path. This flag cannot be used with the ″**-r**″ flag.

-p	Specifies prompt mode. Prompt when moving or removing files.
-r	Removes files that have been filtered by **lppmgr**. **Note:** If the prompt flag is not specified (**-p**), **lppmgr** will remove all filtered files without further user interaction. This flag cannot be used with the ″**-m**″ flag.
-s	Prints space usage info. This will print the amount of space a particular fileset is using and the total amount of space in question. Some buffer space is added for file metadata.
-t	Specifies that **lppmgr** will *not* rebuild the **.toc** file. This can be useful for having a quick look without having to rebuild the entire **.toc** file, which can take some time. Also, this is required for read-only devices.
-u	Causes **lppmgr** to filter for duplicate updates and conflicting updates which are the same level as bases of the same fileset.
-V	Specifies verbose mode. **lppmgr** will give more output in certain situations.
-x	Causes **lppmgr** to filter for superseded updates.
-X	Filters non-system images from a NIM **lpp_source** resource.

Exit Status

0 All **lppmgr** related operations completed successfully.

>0 An error occurred.

Security

Only the root user can execute **lppmgr**.

Examples

1. To list all duplicate and conflicting updates in image source directory **/myimages**, enter:
   ```
   lppmgr -d /myimages -u
   ```
2. To remove all duplicate and conflicting updates in image source directory **/myimages**, enter:
   ```
   lppmgr -d /myimages -u -r
   ```
3. To remove all duplicate and conflicting updates, duplicate base levels, and all message/locale filesets other than ″en_US″ in prompted mode, enter:
   ```
   lppmgr -d /myimages -purb -k en_US
   ```
4. To move all superseded update images and non SIMAGES from NIM **lpp_source** location **/lpps/433** to directory **/backups**, enter:
   ```
   lppmgr -d /lpps/433 -x -X -m  /backups
   ```

Files

/usr/lib/instl/lppmgr	Contains the **lppmgr** command.

Related Information

The **installp** command, **lslpp** command, **lsnim** command.

lpq Command

The **lpq** command includes information for the AIX Print Subsystem **lpq** and the System V Print Subsystem **lpq**.

AIX Print Subsystem lpq Command

Purpose

Examines the spool queue.

Syntax

lpq [**+** [*Number*]] [**-l** | **-W**] [**-P** *Printer*] [*JobNumber*] [*UserName*]

Description

The **lpq** command reports the status of the specified job or all jobs associated with the specified *UserName* and *JobNumber* variables. *JobNumber* variable specifies the number of the job in the spool queue that you want to view. A *UserName* variable specifies viewing the jobs for the name of the person who submitted the job to that queue.

The **lpq** command reports on any jobs currently in the default queue when invoked without any options. Parameters supplied that are not recognized as parameters are interpreted as user names or job numbers to filter out only those jobs of interest.

For each job submitted (each job called by the **lpr** command), the **lpq** command reports the user's name, current rank in the queue, the name of the job, the job identifier (a number that can be supplied to the **lprm** command for removing a specific job), and the total size in blocks. Normally, only as much information as will fit on one line is displayed. Job ordering depends on the algorithm used to scan the spooling directory and is supposed to be FIFO (first-in-first-out). File names making up a job may be unavailable (when the **lpr** command is used as a sink in a pipeline). In this case, the file is indicated as **-** (standard input).

The display generated by the **lpq** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry follows immediately; it contains the client's local queue name (again), followed by the remote queue name. Any jobs submitted to a remote queue are displayed first on the local side and are moved to the remote device as the job is processed on the remote machine.

Since the status commands communicate with remote machines, the status display may occasionally appear to hang while waiting for a response from the remote machine. The command will eventually time out if a connection cannot be established between the two machines.

Flags

-l	Generates the long output format.
+ [*Number*]	Displays the spool queue until it empties. A *Number* variable is the time in seconds before the display regenerates.
-P *Printer*	Displays the spool queue for the printer specified by the *Printer* variable. **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.
-W	Displays a wide version of status information with longer queue names, device names, and job numbers. Longer job number information is available on AIX 4.3.2 and later. This flag cannot be used with the -l flag. If the -l flag and the -W flag are used simultaneously, the first one specified takes precedence.

Examples

1. To display a job number in the print queue `lp0`, enter:

   ```
   lpq  -P lp0
   ```

   This command displays a list similar to the following:

   ```
   Queue    Dev      Status    Job    Files    User      PP    %      Blks    CP    Rnk
   lp0       dlp0    running   39      motd     guest    10    83      12      1     1
   ```

2. To display the status of the default queue in wide format for AIX 4.3.2 or later, enter:

   ```
   lpq  -W
   ```

Files

/usr/bin/lpq	Contains the **lpq** command.
/usr/sbin/qdaemon	Contains the queuing daemon.
/etc/qconfig	Contains the queue configuration file.
/etc/qconfig.bin	Contains the digested, binary version of the **/etc/qconfig** file.
/var/spool/lpd/qdir/*	Contains queue requests.
/var/spool/lpd/stat/*	Contains information on the status of the devices.
/var/spool/qdaemon/*	Contains temporary copies of enqueued files.

Related Information

The **lpr** command, **lprm** command, **lpstat** command, **qchk** command.

The **qconfig** file.

Checking Print Job Status in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Printers, Print Jobs, and Queues in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Spooler Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

System V Print Subsystem lpq Command

Purpose

(BSD) Displays the queue of printer jobs

Syntax

/usr/bin/lpq [**-P***printer*] [-l] [+ [*interval*]] [*job#* ...] [*username* ...]

Description

The **lpq** command displays the contents of a printer queue. It reports the status of jobs specified by *job#*, or all jobs owned by the user specified by *username*. **lpq** reports on all jobs in the default printer queue when invoked with no arguments.

For each print job in the queue, **lpq** reports the user's name, current position, the names of input files comprising the job, the job number (by which it is referred to when using **lprm**) and the total size in bytes. Normally, only as much information as will fit on one line is displayed. Jobs are normally queued on a

first-in-first-out basis. Filenames comprising a job may be unavailable, such as when **lpr** is used at the end of a pipeline; in such cases the filename field indicates the standard input.

If **lpq** warns that there is no daemon present (that is, due to some malfunction), the **lpc** command can be used to restart a printer daemon.

Output formatting is sensitive to the line length of the terminal; this can result in widely-spaced columns.

Flags

-P *printer*
Display information about the queue for the specified *printer*. In the absence of the **-P** flag, the queue to the printer specified by the **PRINTER** variable in the environment is used. If the **PRINTER** variable is not set, the queue for the default printer is used.

-l Display queue information in long format; includes the name of the host from which the job originated.

+[*interval* **]**
Display the spool queue periodically until it empties. This option clears the terminal screen before reporting on the queue. If an *interval* is supplied, **lpq** sleeps that number of seconds in between reports.

Files

/var/spool/lp
spooling directory.

/var/spool/lp/tmp/*system_name/*-0*
request files specifying jobs

Error Codes

lpq: *printer* **is printing**
The **lpq** program queries the spooler **LPSCHED** about the status of the printer. If the printer is disabled, the system administrator can restart the spooler using **lpc**.

lpq: *printer* **waiting for auto-retry (offline ?)**
The daemon could not open the printer device. The printer may be turned off-line. This message can also occur if a printer is out of paper, the paper is jammed, and so on. Another possible cause is that a process, such as an output filter, has exclusive use of the device. The only recourse in this case is to kill the offending process and restart the printer with **lpc**.

lpq: waiting for *host* **to come up**
A daemon is trying to connect to the remote machine named *host*, in order to send the files in the local queue. If the remote machine is up, **lpd** on the remote machine is probably dead or hung and should be restarted using **lpc**.

lpq: sending to *host*
The files are being transferred to the remote *host*, or else the local daemon has hung while trying to transfer the files.

lpq: printer disabled reason:
The printer has been marked as being unavailable with **lpc**.

lpq: The LP print service isn't running or can't be reached.
The **lpsched** process overseeing the spooling queue does not exist. You can restart the printer daemon with **lpc**.

lpq: *printer***: unknown printer**
The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to find the reason.

lpq: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon */usr/lib/lp/lpsched* is running.

lpq: Can't send message to LP print service

lpq: Can't establish contact with LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lpq: Received unexpected message from LP print service
It is likely there is an error in this software. Get help from system administrator.

Related Information
The **lpc** command, **lpr** command, and **lprm** command.

---

lpr Command

The **lpr** command includes information for the AIX Print Subsystem **lpr** and the System V Print Subsystem **lpr**.

AIX Print Subsystem lpr Command

Purpose
Enqueues print jobs.

Syntax
lpr [**-f**] [**-g**] [**-h**] [**-j**] [**-l**] [**-m**] [**-n**] [**-p**] [**-r**] [**-s**] [**-P** *Printer*] [**-#** *NumberCopies*] [**-C** *Class*] [**-J** *Job*] [**-T** *Title*] [**-i** [*NumberColumns*]] [**-w** *Width*] [*File ...*]

Description
The **lpr** command uses a spooling daemon to print the named *File* parameter when facilities become available. If no files are specified, the **lpr** command reads from standard input.

Flags

-# *Number*	Produces multiple copies of output, using the *Number* variable as the number of copies for each file named.
-C *Class*	Specifies the print *Class* as the job classification on the burst page.
-f	Uses a filter that interprets the first character of each line as a standard FORTRAN carriage control character.
-g	The files are assumed to contain standard plot data.
-h	Suppresses printing of the burst page. **Note:** The default is to print a header page and not a trailer page.
-i [*Number*]	Indents output *Number* spaces. If the *Number* variable is not given, eight spaces are used as the default.
-j	Specifies that the message `Job number is: nnn`, where `nnn` is the assigned job number, be displayed to standard output. This occurs only if the job is submitted to a local print queue.
-J *Job*	Prints the *Job* variable as the job name on the burst page. Usually, the **lpr** command uses the name of the first file.
-l	(Lowercase L) Uses a filter that allows control characters to be printed.

-m	Sends mail upon completion of spooling.
-n	Uses a filter that formats files containing *ditroff* (device-independent *troff*) data.
-P *Printer*	Forces output to the *Printer* variable. If this flag is not specified, the following conditions occur: • If a default exists, the **lpr** command uses the default printer. • If the **LPDEST** environment variable is set, then **lpr** uses the value specified by the **LPDEST** variable. If set, this value is always used, even if the **PRINTER** variable is also set. • If the **PRINTER** variable is set and no **LPDEST** variable is set, then **lpr** uses the value specified by the **PRINTER** environment variable. **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.
-p	Uses the **pr** command to format the file (`lpr -p` is very much like `pr | lpr`).
-r	Removes the file upon completion of spooling.
-s	Prints from the files specified on the command line rather than trying to copy them (so large files can be printed). This means the data files should not be modified or removed until they have been printed. Note that this flag only works on the local host (files sent to remote printer hosts are copied anyway), and only with named data files. It does not work if the **lpr** command is at the end of a pipeline.
-T *Title*	Uses the *Title* variable instead of the file name for the title used by the **pr** command.
-w *Number*	Uses the *Number* variable as the page width for the **pr** command.

Examples

1. To print three copies of the files `new.index.c`, `print.index.c`, and `more.c`, enter:

   ```
   lpr  -#3 new.index.c print.index.c more.c
   ```

   Prints three copies of the `new.index.c` file, three copies of the `print.index.c file`, and three copies of the `more.c file`.
2. To print three copies of the concatenation of three files `new.index.c`, `print.index.c`, and `more.c`, enter:

   ```
   cat new.index.c print.index.c more.c | lpr  -#3
   ```
3. To print `Operations` on the burst page, followed by file `new.index.c`, enter:

   ```
   lpr  -C Operations new.index.c
   ```

   This replaces the system name (the name returned by host name) with `Operations` on the burst page.
4. To queue the MyFile file and return the job number, enter:

   ```
   lpr -j MyFile
   ```

Files

/usr/sbin/qdaemon	Queuing daemon.
/etc/qconfig	Queue configuration file.
/etc/qconfig.bin	Digested, binary version of the **/etc/qconfig** file.
/var/spool/lpd/qdir/*	Queue requests.
/var/spool/lpd/stat/*	Information on the status of the queues.
/var/spool/qdaemon	Temporary copies of enqueued files.

Related Information

The **lpd** command, **lpq** command, **lprm** command, **pr** command, **qdaemon** command.

The **qconfig** file.

Starting a Print Job in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Printers, Print Jobs and Queues Overview for Users in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Printer Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

System V Print Subsystem lpr Command

Purpose

(BSD) Sends a job to the printer.

Syntax

/usr/bin/lpr [**-P** *printer*] [**-#** *copies*] [**-C** *class*] [**-J** *job*] [**-T** *title*] [**-i** [*indent*]] [**-w** *cols*] [**-r**] [**-m**] [**-h**] [**-s**] [**-***filter_option*] [*file* ...]

Description

The **lpr** command forwards printer jobs to a spooling area for subsequent printing as facilities become available. Each printer job consists of copies of each *file* you specify. The spool area is managed by the line printer spooler, **lpsched**. **lpr** reads from the standard input if no files are specified.

lp is the preferred interface.

Command-line options cannot be combined into a single argument as with some other commands. The command:

```
lpr -fs
```

is not equivalent to

```
lpr -f -s
```

Placing the **-s** flag first, or writing each option as a separate argument, makes a link as expected.

lpr -p is not precisely equivalent to **pr | lpr**. **lpr -p** puts the current date at the top of each page, rather than the date last modified.

Fonts for **troff** and T[E]X reside on the printer host. It is not possible to use local font libraries.

lpr objects to printing binary files.

If userA uses **su** to become userB and uses */usr/bin/lpr*, then the printer request will be entered as userB, not userA

Flags

-P *printer*
: Send output to the named *printer*. Otherwise send output to the printer named in the **PRINTER** environment variable, or to the default printer, **lp**.

-# *copies*
: Produce the number of *copies* indicated for each named file. For example:

```
lpr -#3 index.c lookup.c
```

produces three copies of *index.c*, followed by three copies of *lookup.c*. On the other hand,

```
cat index.c lookup.c | lpr -#3
```

generates three copies of the concatenation of the files.

-C *class*
Print *class* as the job classification on the burst page. For example,

```
lpr -C Operations new.index.c
```

replaces the system name (the name returned by ``hostname'') with **Operations** on the burst page, and prints the file *new.index.c*.

-J *job* Print *job* as the job name on the burst page. Usually, **lpr** uses the first file's name.

-T *title* Use *title* instead of the file name for the title used by **pr**.

-i[*indent***]**
Indent output *indent* <Space> characters. Eight <Space> characters is the default.

-w *cols*
Use *cols* as the page width for **pr**.

-r Remove the file upon completion of spooling, or upon completion of printing with the **-s** flag.

-m Send mail upon completion.

-h Suppress printing the burst page.

-s Use the full pathnames (not symbolic links) of the files to be printed rather than trying to copy them. This means the data files should not be modified or removed until they have been printed. This flag only prevents copies of local files from being made. Jobs from remote hosts are copied anyway. The **-s** flag only works with named data files; if the **lpr** command is at the end of a pipeline, the data is copied to the spool.

filter_option
The following single letter options notify the line printer spooler that the files are not standard text files. The spooling daemon will use the appropriate filters to print the data accordingly.

-p Use **pr** to format the files (**lpr -p** is very much like **pr | lpr**).

-l Print control characters and suppress page breaks.

-t The files contain **troff** (cat phototypesetter) binary data.

-n The files contain data from *ditroff* (device independent **troff**).

-d The files contain data from *tex* (DVI format from Stanford).

-g The files contain standard plot data as produced by the routine **plot** for the filters used by the printer spooler.

-v The files contain a raster image. The printer must support an appropriate imaging model such as PostScript in order to print the image.

-c The files contain data produced by *cifplot*.

-f Interpret the first character of each line as a standard FORTRAN carriage control character.

If no *filter_option* is given (and the printer can interpret PostScript), the string `**%!**' as the first two characters of a file indicates that it contains PostScript commands.

These filter options offer a standard user interface, and all options may not be available for, nor applicable to, all printers.

Files

/usr/lib/lp/lpsched
System V line printer spooler

/var/spool/lp/tmp/*
directories used for spooling

/var/spool/lp/tmp/*system/*-0*
spooler control files

/var/spool/lp/tmp/*system/*-N*
(*N* is an integer and > 0) data files specified in `*-0' files

Error Codes

lpr: *printer***: unknown printer**
The *printer* was not found in the LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to find the reason.

lpr: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lpsched** is running.

lpr: *printer***: printer queue is disabled**
This means the queue was turned off with

```
/usr/etc/lpc disable printer
```

to prevent **lpr** from putting files in the queue. This is usually done by the system manager when a printer is going to be down for a long time. The printer can be turned back on by a privileged user with **lpc**.

lpr: Can't send message to the LP print service

lpr: Can't establish contact with the LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lpr: Received unexpected message from LP print service
It is likely there is an error in this software. Get help from system administrator.

lpr: There is no filter to convert the file content
Use the **lpstat -p -l** command to find a printer that can handle the file type directly, or consult with your system administrator.

lpr: cannot access the file
Make sure file names are valid.

Related Information

The **lpc** command, **lpq** command, **lprm** command, and **troff** command.

---

lprm Command

The **lprm** command includes information for the AIX Print Subsystem **lprm** and the System V Print Subsystem **lprm**.

AIX Print Subsystem lprm Command

Purpose
Removes jobs from the line printer spooling queue.

Syntax
lprm [**-P** *Printer*] [*JobNumber*] [*UserName ...*] [**-**]

Description
The **lprm** command removes one or more jobs from the spool queue of a printer.

You cannot run the **lprm** command without specifying a job number, the **-** (minus sign) flag, or at least one user name.

Specifying a *UserName* parameter, or list of names, causes the **lprm** command to attempt to remove any jobs queued belonging to that user (or users).

You can remove an individual job from a queue by specifying its *JobNumber*. This job number is obtained by using the **lpq** command.

Flags

- Removes all jobs a user owns. Someone with root user authority can use this flag to remove all jobs from a queue. This flag is not valid for remote print.

-P *Printer* Specifies the queue associated with a specific *Printer* variable. If this flag is not specified, the following conditions occur:

- If the **LPDEST** environment variable is set, then **lprm** uses the value specified by the **LPDEST** variable. If set, this value is always used, even if the **PRINTER** variable is also set.
- If the **PRINTER** variable is set and no **LPDEST** variable is set, then **lprm** uses the value specified by the **PRINTER** environment variable.

If neither the **LPDEST** nor the **PRINTER** variable is set, the **lprm** command removes jobs from the default queue.

> **Note:** Any destination command line options override both the **LPDEST** and the **PRINTER** environment variables.

Examples
1. To remove job number `13` from the default printer queue, enter:
   ```
   lprm 13
   ```
2. To remove job number `13` from printer queue `lp0`, enter:
   ```
   lprm  -P lp0 13
   ```
3. To remove a job from the printer queue for a certain user, enter:
   ```
   lprm guest
   ```

Files

/usr/bin/lprm	Contains the **lprm** command.
/etc/qconfig	Contains the configuration file.

Related Information
The **lpq** command, **lpr** command, **qcan** command.

Canceling a Print Job (qcan Command) in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

System V Print Subsystem lprm Command

Purpose

(BSD) Remove jobs from the printer queue

Syntax

/usr/bin/lprm [**-P***printer*] [-] [*job #* ...] [*username*...]

Description

The **lprm** command removes a job or jobs from a printer's spooling queue. Since the spool directory is protected from users, using **lprm** is normally the only method by which a user can remove a job.

Without any arguments, **lprm** deletes the job that is currently active, provided that the user who invoked **lprm** owns that job.

When the privileged user specifies a *username*, **lprm** removes all jobs belonging to that user.

You can remove a specific job by supplying its job number as an argument, which you can obtain using **lpq**. For example:

```
lpq  -Phost
host is ready and printing
Rank         Owner   Job     Files           Total Size
active       wendy   385     standard input  35501 bytes
lprm -Phost 385
```

lprm reports the names of any files it removes, and is silent if there are no applicable jobs to remove.

lprm Sends the request to cancel a job to the print spooler, **LPSCHED**.

An active job may be incorrectly identified for removal by an **lprm** command issued with no arguments. During the interval between an **lpq** command and the execution of **lprm**, the next job in queue may have become active; that job may be removed unintentionally if it is owned by you. To avoid this, supply **lprm** with the job number to remove when a critical job that you own is next in line.

Only the privileged user can remove print jobs submitted from another host.

Flags

-P*printer*
: Specify the queue associated with a specific printer. Otherwise the value of the **PRINTER** variable in the environment is used. If this variable is unset, the queue for the default printer is used.

-
: Remove all jobs owned by you. If invoked by the privileged user, all jobs in the spool are removed. Job ownership is determined by the user's login name and host name on the machine where the **lpr** command was executed.

Files

/var/spool/lp/*
: spooling directories

Error Codes

lprm: *printer***: unknown printer**
The *printer* was not found in the System V LP database. Usually this is a typing mistake; however, it may indicate that the printer does not exist on the system. Use **lpstat -p** to get the status of printers.

lprm: error on opening queue to spooler
The connection to **lpsched** on the local machine failed. This usually means the printer server started at boot time has died or is hung. Check if the printer spooler daemon **/usr/lib/lp/lpsched** is running.

lprm: Can't send message to the LP print service

lprm: Can't receive message from the LP print service
These indicate that the LP print service has been stopped. Get help from the system administrator.

lprm: Received unexpected message from the LP print service
It is likely there is an error in this software. Get help from system administrator.

lprm: Can't cancel request
You are not allowed to remove another user's print request.

Related Information

The **lp** command, **lpq** command, **lpr** command, and **lpsched** command.

---

lpsched Command

Note: This is a System V Print Subsystem command.

Purpose

Starts/stops the print service.

Syntax

/usr/lib/lp/lpsched

lpshut

Description

The **lpsched** command starts the LP print service.

The **lpshut** command shuts down the print service. All printers that are printing at the time the **lpshut** command is invoked stop printing. When **lpsched** is started again, requests that were printing at the time a printer was shut down are reprinted from the beginning.

You must have the appropriate privilege to run these commands.

If the scheduler fails to run, check the **lpsched** log file, which contains all failed attempts to load print requests, printer descriptions, forms, filters, classes, alerts, and systems. The log files are located in **/var/lp/logs**. Useful information on the networked print service can also be found in the **/var/lp/logs/lpNet** log file.

Files

/var/spool/lp/*

Related Information

The **accept** command, **enable** command, **lpadmin** command.

---

lpstat Command

The **lpstat** command includes information for the AIX Print Subsystem **lpstat** and the System V Print Subsystem **lpstat**.

AIX Print Subsystem lpstat Command

Purpose

Displays line printer status information.

Syntax

lpstat [**-a***List*] [**-c***List*] [**-d**] [**-o***List*] [**-p***List*] [**-r**] [**-s**] [**-t**] [**-u***List*] [**-v***List*] [**-W**]

Description

The **lpstat** command displays information about the current status of the line printer.

If no flags are given, **lpstat** prints the status of all requests made by the **lp** command.

Flags can appear in any order and can be repeated. Some flags take an optional list as a parameter. Enter the list as either a list of items separated by commas, as in `lpstat -aQueue1,Queue2`, or as a list of items enclosed in single or double quotes and separated either by commas or one or more spaces, as in, for example, `lpstat -a"Queue1 Queue2"` or `lpstat -a"Queue1,Queue2"` or `lpstat -a'Queue1 Queue2'` or `lpstat -a'Queue1,Queue2'`.

If you specify a flag with no parameters, all information pertaining to that flag is printed.

The display generated by the **lpstat** command contains two entries for remote queues. The first entry contains the client's local queue and local device name and its status information. The second entry contains the client's local queue name followed by the remote queue name. The spooling subsystem first displays remote print requests on the local queue. When the remote machine begins to process the remote print job, the status display for the print job moves to the remote queue.

When a status command communicates with a remote host, the display occasionally appears to hang while the command waits for a response from the remote machine. The command eventually times out if no connection is established between the two machines.

Flags

-a*List* Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q -P***Queue1* **-P***Queue2* ... command (where *Queue1*, *Queue2*, etc., are items in *List*).

-c*List* Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q -P***Queue1* **-P***Queue2* ... command (where *Queue1*, *Queue2*, etc., are items in *List*).

-d Prints the status information for the system default destination for the **lp** command. Specifying the **lpstat** command with this flag is the same as specifying the **enq -q** command.

-o*List* Prints the status of print requests or print queues. *List* is a list of intermixed printer names and job numbers.

-p*List* Prints the status of printers.

-r Provides status and job information on queues. Specifying the **lpstat** command with this flag is the same as specifying the **enq -A** command.

-s Displays a status summary, including a list of printers and their associated devices. Specifying the **lpstat** command with this flag is the same as specifying the **enq -A** command.

-t Displays all status information, including a list of printers and their associated devices. Specifying the **lpstat** command with this flag is the same as specifying the **enq -AL** command.

-u*List* Prints the status of all print requests for users specified in *List*. *List* is a list of login names. Specifying the **lpstat** command with this flag is the same as specifying the **enq -u** *UserName* command.

-v*List* Prints the status of printers. The *List* variable is a list of printer names.

-W Displays a wide version of status information with longer queue names, device names, and job numbers. Longer job number information is available on AIX 4.3.2 and later. This flag cannot be used with the **-t** flag. If the **-t** flag and the **-W** flag are used simultaneously, the first one specified takes precedence.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To display the status for all print queues, enter:

   ```
   lpstat
   ```

2. To display the long status for all printers, enter:

   ```
   lpstat -t
   ```

3. To display a job number in the print queue `lp0`, enter:

   ```
   lpstat -plp0
   ```

   This command displays a list similar to the following:

   ```
   Queue  Dev      Status      Job     Files      User      PP      %      Blks      CP      Rnk
   lp0    dlp0     running     39      motd       guest     10      83      12        1       1
   ```

4. To display the status for users `root`, `ghandi`, and `king`, enter:

   ```
   lpstat -u"root,ghandi,king"
   ```

5. To display the status of all print queues in wide format for AIX 4.3.2 or later, enter:

   ```
   lpstat -W
   ```

Files

/var/spool/lpd/* Contains temporary copies of remote enqueued files.

Related Information

The **disable** command, **enable** command, **enq** command, **lp** command, **lpr** command, **qchk** command.

Checking Print Job Status in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Printers, Print Jobs and Queues Overview for Users in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Queuing System Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

System V Print Subsystem lpstat Command

Purpose

Prints information about the status of the LP print service.

Syntax

lpstat [*flags*] [*request-ID-list*]

Description

The **lpstat** command displays information about the current status of the LP print service. If no *flags* are given, **lpstat** displays the status of all print requests made by you. (See **lp** for more information.)

The command **lpstat -o** *printername* is used to list all the requests queued on the specified printer. If *printername* points to a remote printer (see **lpadmin**), then **lpstat -o** *printername* lists all the requests on the remote printer, not just those submitted locally.

Any arguments that are not *flags* are assumed to be *request-IDs* as returned by **lp**. The **lpstat** command displays the status of such requests. The *flags* may appear in any order and may be repeated and intermixed with other arguments. Some of the keyletters below may be followed by an optional *list* that can be in one of two forms:

- a list of items separated by commas, for example, **-p** *printer1*,*printer2*
- a list of items separated by spaces and enclosed in quotes, for example, **-u** *"user1 user2 user3"*

Specifying **all** after any keyletter that takes *list* as an argument causes all information relevant to the keyletter to be displayed. For example, the command **lpstat -a all** lists the accepting status of all print destinations.

The omission of a *list* following such keyletters causes all information relevant to the keyletter to be displayed. For example, the command **lpstat -a** is equivalent to **lpstat -a all**.

There are two exceptions to the behavior of the **all** keyword. The first is when it is used in conjunction with the **-o** flag: **lpstat -o all** only lists requests submitted locally to remote printers. The second is when it is used with directory-enabled print queues. Use of the **all** keyword will only return non-directory-enabled print queues. **lpstat -a** *list* will report whether the both directory-enabled and non-directory-enabled print queues in *list* are accepting requests. For the **-a** and **-b** flags, **lpsched** will remember the directory-enabled print queues specified until it is restarted. Subsequent calls to **lpstat -a** and **lpstat -p** will report the status of all non-directory-enabled print queues as well as the directory-enabled print queues previously specified. Once **lpsched** has been restarted, the use of the **all** keyword with the **lpstat** command will once again only display non-directory-enabled print queues. The **dslpsearch** command should be used to search for defined directory-enabled print queues.

Flags

-a [*list***]**
: Report whether print destinations are accepting requests. *list* is a list of intermixed printer names and class names.

-c [*list***]**
: Report names of all classes and their members. *list* is a list of class names.

-d Report what the system default destination is (if any).

-f [*list***] [-l]**
Verify that the forms in *list* are recognized by the LP print service. *list* is a list of forms; the default is **all**. The **-l** option will list the form parameters.

-o [*list***] [-l]**
Report the status of print requests. *list* is a list of intermixed printer names, class names, and *request-IDs*. The keyletter **-o** may be omitted. The **-l** option lists for each request whether it is queued for, assigned to, or being printed on a local printer, the form required (if any), and the character set or print wheel required (if any). Note that required forms (if any) are not listed for remote printers.

-p [*list***] [-D] [-l]**
If the **-D** flag is given, a brief description is printed for each printer in *list*. If the **-l** flag is given, a full description of each printer's configuration is given, including the form mounted, the acceptable content and printer types, a printer description, the interface used, and so on.

In order to maintain system security access information, the information needed to produce the printer status given by **lpstat -p** is available only if the LP scheduler is running.

-r Report the status of the LP request scheduler (whether it is running).

-R Report a number showing the rank order of jobs in the print queue for each printer.

-s [-l] Display a status summary, including the status of the LP scheduler, the system default destination, a list of class names and their members, a list of printers and their associated devices, a list of the systems sharing print services, a list of all forms and their availability, and a list of all recognized character sets and print wheels. The **-l** flag displays all parameters for each form and the printer name where each character set or print wheel is available.

-S [*list***] [-l]**
Verify that the character sets or the print wheels specified in *list* are recognized by the LP print service. Items in *list* can be character sets or print wheels; the default for *list* is **all**. If the **-l** flag is given, each line is appended by a list of printers that can handle the print wheel or character set. The list also shows whether the print wheel or character set is mounted or specifies the built-in character set into which it maps.

-t [-l] Display all status information: all the information obtained with the **-s** flag, plus the acceptance and idle/busy status of all printers and status of all requests. The **-l** flag displays more detail as described for the **-f**, **-o**, **-p**, and **-s** flag.

-u [*login-ID-list***]**
Display the status of output requests for users. The *login-ID-list* argument may include any or all of the following constructs:

login-ID
a user on the local system

*system-name***!***login-ID*
a user on system *system-name*

*system-name***!all**
all users on system *system-name*

all!*login-ID*
a user on all systems

all all users on the local system

all!all all users on all systems

The default value of *login-ID-list* is **all**.

lpsystem Command

Note: This is a System V Print Subsystem command.

Purpose

Registers remote systems with the print service.

Syntax

lpsystem [**-T** *Timeout*] [**-R** *Retry*] [**-y** *Comment*] *SystemName* [*SystemName* . . .]

lpsystem -l [*SystemName* . . .]

lpsystem -r *SystemName* [*SystemName* . . .]

lpsystem -A

Description

The **lpsystem** command defines parameters for the LP print service, with respect to communication (via a high-speed network such as TCP/IP) with remote systems.

Specifically, the **lpsystem** command defines remote systems with which the local LP print service can exchange print requests. These remote systems are described to the local LP print service in terms of several parameters that control communication: type, retry, and timeout. These parameters are defined in **/etc/lp/Systems**. You can edit this file with a text editor (such as **vi**), but editing is not recommended. By using **lpsystem**, you can ensure that **lpsched** is notified of any changes to the *Systems* file.

The *Timeout* parameter specifies the length of time (in minutes) that the print service should allow a network connection to be idle. If the connection to the remote system is idle (that is, there is no network traffic) for *N* minutes, then drop the connection. (When there is more work, the connection is re-established.) Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0. If a decimal number is used for *N*, it is truncated to the whole number. The value **n** means never time out; **0** means as soon as the connection is idle, drop it. The default is **n**.

The *Retry* parameter specifies the length of time (in minutes) to wait before trying to re-establish a connection to the remote system, when the connection was dropped abnormally (that is, a network error). Legal values are **n**, **0**, and *N*, where *N* is an integer greater than 0. It means wait *N* minutes before trying to reconnect. If a decimal number is used for *N*, it is truncated to the whole number. (The default is 10 minutes.) The value **n** means do not retry dropped connections until there is more work; **0** means try to reconnect immediately.

The *Comment* parameter allows you to associate a free form comment with the system entry. This is visible when **lpsystem -l** is used.

The *SystemName* is the name of the remote system from which you want to be able to receive jobs and to which you want to be able to send jobs. A special entry is provided with the **/etc/lp/Systems** file by default, which allows all connections to **bsd** systems. That entry uses the asterisk (*) as the *SystemName*.

The command **lpsystem -l** [*SystemName*] prints out a description of the parameters associated with *SystemName* (if a system has been specified) or with all the systems in its database (if *SystemName* has not been specified).

The command **lpsystem -r** *SystemName* removes the entry associated with *SystemName*. The print service no longer accepts jobs from that system or send jobs to it, even if the remote printer is still defined

on the local system. The scheduler must be running when the removal of a systems file entry occurs, because the scheduler checks whether the system entry is currently used by a printer destination. If currently used, the system entry cannot be removed.

If you use **lpsystem -r** *SystemName* to remove a system and you have active printers for that system, you will not be allowed to remove the system from the system file. **lpsystem -r** *SystemName* only works if no printers for that system exist.

With respect to the semantics of the *Timeout* and *Retry* values, the print service uses one process for each remote system with which it communicates, and it communicates with a remote system only when there is work to be done on that system or work is being sent from that system.

The system initiating the connection is the master process, and the system accepting the connection is the secondary process. This designation serves only to determine which process dies (the secondary) when a connection is dropped. This helps prevent more than one process communicating with a remote system. All connections are bi-directional, regardless of the master/secondary designation. You cannot control a system's master/secondary designation. Typically, a client machine has the master child, and the server machine has the secondary child. If a master process times out, then both the secondary and master exit. If a secondary process times out, then it is possible that the master may still live and retry the connection after the retry interval. Therefore, one system's resource management strategy can affect another system's strategy.

All forms of the **lpsystem** command accept ***** (asterisk enclosed in double quotes) for *SystemName*.

Depending upon the configuration of the name server, you may need to change the entry in the *SystemName* field in **/etc/lp/Systems** to a full domain name.

Flags

-A	Prints out the TCP/IP address in a format.
-l [*SystemName*]	Prints out a description of the parameters associated with *SystemName* or with all the systems in its database.
-r *SystemName*	Removes the entry associated with *SystemName*.
-R *Retry*	Specifies time to wait before trying to reestablish a connection for a remote system.
-T *Timeout*	Specifies the time allowed for a network connection to be idle. *Timeout* is in minutes. Default is to never time out.
-y *Comment*	The *Comment* parameter allows you to associate a free-form comment with the system entry.

Security

Only a user with appropriate privileges may execute the **lpsystem** command.

Files

/etc/lp/*
/var/spool/lp/*

Related Information

The **lpsched** command.

lptest Command

Purpose
Generates the line printer ripple pattern.

Syntax
lptest [;*Length Count*]

Description
The **lptest** command writes the traditional ″ripple″ test pattern on a standard output device such as a terminal or a printer. In 96 lines, this pattern will print all 96 printable ASCII characters in each position. While originally created to test printers, the ripple pattern is quite useful for testing terminals, driving terminal ports for debug purposes, or any other task where a quick supply of random data is needed.

Using the **lptest** command, you can specify the output line length if the default length of 79 is not appropriate. You can also specify the number of output lines to be generated if the default *Count* parameter of 200 is not appropriate. Note that if *Count* parameter is specified, *Length* must also be specified.

Examples
To display or print `100` lines of `80`-column test output to standard output, enter:

```
lptest 80 100
```

Related Information
The **cancel** command, **disable** command, **enable** command, **lp** command, **lpstat** command.

lpusers Command

Note: This is a System V Print Subsystem command.

Purpose
Set printing queue priorities.

Syntax
lpusers -d *PriorityLevel*

lpusers -q *PriorityLimit* **-u** *LoginIDList*

lpusers -u *LoginIDList*

lpusers -q *PriorityLimit*

lpusers -l

Description
The **lpusers** command sets limits to the queue priority level that can be assigned to jobs submitted by users of the LP print service.

The first form of the command (with **-d**) sets the system-wide priority default to *PriorityLevel*, where *PriorityLevel* is a value of 0 to 39, with 0 being the highest priority. If a user does not specify a priority level with a print request (see the **lp** command), the default priority is used. Initially, the default priority level is 20.

The second form of the command (with **-q** and **-u**) sets the default *PriorityLimit* (from 0 to 39) that the users in the *LoginIDList* can request when submitting a print request. The *LoginIDList* parameter may include any or all of the following constructs:

Users that have been given a limit cannot submit a print request with a higher priority level than the one assigned, nor can they change a request already submitted to have a higher priority. Any print requests submitted with priority levels higher than allowed will be given the highest priority allowed.

The third form of the command (with **-u**) removes any explicit priority limit for the specified users.

The fourth form of the command (with **-q**) sets the default priority limit for all users not explicitly covered by the use of the second form of this command.

The last form of the command (with **-l**) lists the default priority level and the priority limits assigned to users.

Parameters

LoginID	Specifies a a user on the local system.
*system_name***!***login-ID*	User on the system *system_name*
*system_name***!all**	Users on system *system_name*
all!*login-ID*	User on all systems
all	Users on the local system

Flags

-d *PriorityLevel*	Sets the system-wide priority default to *PriorityLevel*.
-l	Lists the default priority level and the priority limits assigned to users.
-q *PriorityLimit*	Sets the default highest priority level for all users not explicitly covered.
-q *PriorityLimit* **-u** *LoginIDList*	Sets the default highest priority level users in *LoginIDList* can request when submitting a print request.
-u *LoginIDList*	Removes any explicit priority level for the specified users.

Related Information

The **lp** command.

---

ls Command

Purpose

Displays the contents of a directory.

Syntax

To Display Contents of Directory or Name of File

ls [**-1**] [**-A**] [**-C**] [**-F**] [**-H** | **-L**] [**-N**] [**-R**] [**-a**] [**-b**] [**-c**] [**-d**] [**-e**] [**-f**] [**-g**] [**-i**] [**-l**] [**-m**] [**-n**] [**-o**] [**-p**] [**-q**] [**-r**] [**-s**] [**-t**] [**-u**] [**-x**] [*File* ...]

To Display Contents of Directory

ls -f [**-C**] [**-d**] [**-i**] [**-m**] [**-s**] [**-x**] [**-1**] [*Directory ...*]

Description

The **ls** command writes to standard output the contents of each specified *Directory* parameter or the name of each specified *File* parameter, along with any other information you ask for with the flags. If you do not specify a *File* or *Directory* parameter, the **ls** command displays the contents of the current directory.

Specifying more than one of the options in the mutually exclusive pairs is not considered an error. The last option specified in each pair determines the output format.

By default, the **ls** command displays all information in alphabetic order by file name. The collating sequence is determined by the **LANG** or **LC_COLLATE** environment variable. The ″National Language Support Overview for Programming″ in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs* contains more details.

When the **ls** command displays the contents of a directory, it does not show entries for files whose names begin with a **.** (dot) unless you use the **-a** or **-A** flag. If the command is executed by root, it uses the **-A** flag by default.

There are three main ways to format the output:

- List one entry per line.
- List entries in multiple columns by specifying either the **-C** or **-x** flag. The **-C** flag is the default format when output is to a tty. The **ls** command displays single column output if file or directory names are too long.
- List entries in a comma-separated series by specifying the **-m** flag.

To determine the number of character positions in the output line, the **ls** command uses the **COLUMNS** environment variable. If this variable is not set, the command gets the current column value of the display. If the **ls** command cannot determine the number of character positions by either of these methods, it uses a default value of 80.

The mode displayed with the **-e** and **-l** flags is interpreted as follows:

If the first character is:

d The entry is a directory.
b The entry is a block special file.
c The entry is a character special file.
l The entry is a symbolic link, and either the **-N** flag was specified or the symbolic link did not point to an existing file.
p The entry is a first-in,first-out (FIFO) special file.
s The entry is a local socket.
- The entry is an ordinary file.

The next nine characters are divided into three sets of three characters each. The first set of three characters show the owner's permission. The next set of three characters show the permission of the other users in the group. The last set of three characters shows the permission of anyone else with access to the file. The three characters in each set indicate, respectively, read, write, and execute permission of the file. Execute permission of a directory lets you search a directory for a specified file.

Permissions are indicated as follows:

r Read

w Write (edit)
x Execute (search)
- Corresponding permission not granted

The group-execute permission character is **s** if the file has set-group-ID mode. The user-execute permission character is **s** if the file has set-user-ID mode. The last character of the mode (usually **x** or **-**) is **T** if the 01000 (octal) bit of the mode is set (see the **chmod** command for the meaning of this mode). The indications of set-ID and 01000 bit of the mode are capitalized (**S** and **T**, respectively) if the corresponding execute permission is not set. The mode **t** indicates that the sticky bit is on for the file or the directory.

The mode displayed with the **-e** flag is the same as with the **-l** flag, except for the addition of an 11th character interpreted as follows:

+ Indicates a file has extended security information. For example, the file may have extended **ACL**, **TCB**, or **TP** attributes in the mode.

The access control information (**ACL**) of a file is displayed by using the **aclget** command. The value of the **TCB** and **TP** attributes are displayed by using the **chtcb** command.

- Indicates a file does not have extended security information.

When the size of the files in a directory are listed, the **ls** command displays a total count of blocks, including indirect blocks.

Flags

-A Lists all entries except **.** (dot) and **..** (dot-dot).
-a Lists all entries in the directory, including the entries that begin with a **.** (dot).
-b Displays nonprintable characters in an octal (\n*nn*) notation.
-c Uses the time of last modification of the i-node for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag must be used with either the **-t** or **-l** flag, or both.
-C Sorts output vertically in a multicolumn format. This is the default method when output is to a terminal.
-d Displays only the information for the directory named. Directories are treated like files, which is helpful when using the **-l** flag to get the status of a directory.
-e Displays the mode (including security information), number of links, owner, group, size (in bytes), time of last modification, and name of each file. If the file is a special file, the size field contains the major and minor device numbers. If the file is a symbolic link, the path name of the linked-to file is printed preceded by a -> (minus, greater than) sign. The attributes of the symbolic link are displayed.
-E Lists space reservation, fixed extent size, and extent allocation flag information for a file. **-l** must be specified with this flag.
-f Lists the name in each slot for each directory specified in the *Directory* parameter. This flag turns off the **-l**, **-t**, **-s**, and **-r** flags, and turns on the **-a** flag. The order of the listing is the order in which entries appear in the directory.
-F Puts a / (slash) after each file name if the file is a directory, an * (asterisk) if the file can be executed, an = (equal sign) if the file is a socket, a | (pipe) sign if the file is a FIFO, and an @ for a symbolic link.

Note: Symbolic links are displayed with the trailing -> only if the **-N** flag is used or if the link points to a nonexistent file. Otherwise, information about the target file is displayed. You can also invoke this option by entering the **ls -f** command.

-g Displays the same information as the **-l** flag, except the **-g** flag suppresses display of the owner and symbolic link information.
-H If a symbolic link referencing a file of type directory is specified on the command line, the **ls** command shall evaluate the file information and file type to be those of the file referenced by the link, and not the link itself; however, the **ls** command shall write the name of the link itself and not the file referenced by the link.
-i Displays the i-node number in the first column of the report for each file.

-L Lists the file or directory contents that the link references. This is the default action. Symbolic links are followed. If the **-l** option is used, the **-N** option becomes the default, and no symbolic links are followed. When the **-l** option is used, only the **-L** option can override the **-N** default.

-l (Lower case L) Displays the mode, number of links, owner, group, size (in bytes), and time of last modification for each file. If the file is a special file, the size field contains the major and minor device numbers. If the time of last modification is greater than six months ago, the time field is shown in the format **month date year** where as files modified within six months the time field is shown as **month date time** format.

If the file is a symbolic link, the path name of the linked-to file is printed preceded by a ->. The attributes of the symbolic link are displayed. The **-n**, **-g**, and **-o** flag overrides the **-l** flag.

Notes:

1. A symbolically linked file is followed by an arrow and the contents of the symbolic link.
2. The performance of the **ls** command when used with the **-l** option can be improved by executing the **mkpasswd** command. This is helpful when a directory contains files owned by different users, such as the **/tmp** directory.

-m Uses stream output format (a comma-separated series).

-n Displays the same information as the **-l** flag, except that the **-n** flag displays the user and the group IDs instead of the user and group names.

-N Does not follow symbolic links when determining the status of a file.

Note: If both the **-L** and **-N** options are used, the last one will dominate. Also, any time a symbolic link is given that includes a / (slash) as the final character, the link will automatically be followed regardless of any options used.

-o Displays the same information as the **-l** flag, except the **-o** flag suppresses display of the group and symbolic link information.

-p Puts a slash after each file name if that file is a directory. This is useful when you pipe the output of the **ls** command to the **pr** command, as follows:

```
ls -p | pr -5 -t -w80
```

-q Displays nonprintable characters in file names as a ? (question mark).

-r Reverses the order of the sort, giving reverse alphabetic or the oldest first, as appropriate.

-R Lists all subdirectories recursively.

-s Gives size in kilobytes (including indirect blocks) for each entry.

-t Sorts by time of last modification (latest first) instead of by name.

-u Uses the time of the last access, instead of the time of the last modification, for either sorting (when used with the **-t** flag) or for displaying (when used with the **-l** flag). This flag has no effect if it is not used with either the **-t** or **-l** flag, or both.

-x Sorts output horizontally in a multicolumn format.

-1 Forces output into one-entry-per-line format. This is the default when the output is not directed to a terminal.

Exit Status

This command returns the following exit values:

0 All files were written successfully.

>0 An error occurred.

Examples

1. To list all files in the current directory, type:

   ```
   ls -a
   ```

   This lists all files, including **.** (dot), **..** (dot-dot), and other files with names beginning with a dot.
2. To display detailed information, type:

```
ls   -l chap1  .profile
```

This displays a long listing with detailed information about `chap1` and `.profile`.

3. To display detailed information about a directory, type:

```
ls   -d   -l  .  manual  manual/chap1
```

This displays a long listing for the directories `.` and `manual`, and for the file `manual/chap1`. Without the **-d** flag, this would list the files in the `.` and `manual` directories instead of the detailed information about the directories themselves.

4. To list the files in order of modification time, type:

```
ls   -l   -t
```

This displays a long listing of the files that were modified most recently, followed by the older files.

Files

/usr/bin/ls	Contains the **ls** command.
/etc/passwd	Contains user IDs.
/etc/group	Contains group IDs.
/usr/share/lib/terminfo/*	Contains terminal information.

Related Information

The **aclget** command, **chmod** command, **chtcb** command, **find** command, **mkpasswd** command, **qprt** command.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes the structure and characteristics of directories in the file system.

Files and Directories Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes files, file types, and how to name files.

Linking Files and Directories in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* explains the concept of file linking.

Shells Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* describes shells, the different types, and how they affect the way commands are interpreted.

File and Directory Access Modes in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* introduces file ownership and permissions to access files and directories.

National Language Support Overview for Programming in *AIX 5L Version 5.2 National Language Support Guide and Reference* explains collating sequences, equivalence classes, and locale.

ls-secldapclntd Command

Purpose

The **ls-secldapclntd** command lists the status of the **secldapclntd** daemon process.

Syntax

/usr/sbin/ls-secldapclntd

Description

The **ls-secldapclntd** command lists the **secldapclntd** daemon status. The information returned includes the following:

- The LDAP server the **secldapclntd** daemon is talking to
- The LDAP server port number
- The version of the LDAP protocol used
- User base DN
- Group base DN
- System (id) base DN
- User cache size
- User cache size used
- Group cache size
- Group cache size used
- Cache time out (time to live) value
- **secldapclntd** to LDAP server heart beat interval
- Number of thread used by **secldapclntd** daemon
- User objectclass used in the LDAP server
- Group objectclass used in the LDAP server

Example

1. 1.To list the status of the **secldapclntd** daemon, type:

```
/usr/sbin/ls-secldapclntd
```

Files

/etc/security/ldap/ldap.cfg Contains information needed by the **secldapclntd** daemon to connect to the server.

Related Information

The **secldapclntd** daemon

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

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

---

lsactdef Command

Purpose

Lists (displays) action definitions of a resource or resource class.

Syntax

lsactdef [**–h**] [**–c**] [**–p** *property*] [**–s i | o**] [**–e**] [**–l | –i | –t | –d | –D** *delimiter*] [**–x**] [**–TV**] [*resource_class* [*action*...]]

Description

The **lsactdef** command lists the action definitions of a resource or resource class. Use **lsactdef** with no parameters specified to get a list of all resource class names.

By default, the resource action definitions are displayed. To see the resource class action definitions, specify the **-c** flag.

By default, when no actions are specified on the command line, only actions that are defined as public are displayed. To override this default, use the **-p** flag or specify on the command line the names of the actions whose definition you wish to display.

To see the structured data definition that is required as input when this action is invoked, specify the **-s i** flag. To see the structured data definition linked with the output that results from invoking this action, specify the **-s o** flag.

By default, for the actions that contain descriptions, the descriptions are not displayed. Specify the **-e** flag to display the descriptions. Because the translation of these descriptions may take some time and because some of the descriptions are very long, the default is not to display these descriptions.

Flags

–c Displays the resource class action definitions. By default the resource action definitions are displayed.

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you wish to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify a delimiter other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–e Expands the descriptions. By default the definitions are displayed without the textual descriptions because they can be lengthy. Specify this flag to see both the definitions and descriptions.

–i Specifies input format. Generates a template of the **Resource_Data_Input** file. The output is displayed in long (stanza) format. The attribute's SD element data types are displayed as the value in the *Attr=value* pairs. It is suggested that when you use this flag, the output of the **lsactdef** command be directed to a file. This flag overrides the **–s o** flag.

–l Specifies long formatted output, one entry per line. This is the default display format. If the **lsactdef** command is issued with the **-l** flag, but without a resource class name, the **-l** flag is ignored when the command returns the list of defined resource class names.

–p *property*
Displays actions with the specified *property*. By default, only the definitions for public actions are displayed. To display all action definitions regardless of the action property, use the **-p 0** flag.

Action properties:

0x0001 Long running

0x0002 Public

A decimal or hexadecimal value can be specified for the property. To request the action definitions for all actions that have one or more properties, ″OR″ the properties of interest together and then specify the ″ORed″ value with the **-p** flag. For example, to request the action definitions for all actions that are long running or public, enter:

```
-p 0x03
```

–s i | o
Displays the structured data definition for the action input or action response.

i Displays the action input structured data definitions. This is the default.

o Displays the action response (output) structured data definitions.

–t Specifies tabular formatted output. Each attribute is displayed in a separate column, one resource per line.

–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.

–x Suppresses header printing.

Parameters

action If a *resource_class* parameter is specified, zero or more action names may be specified. If no *action* parameter is specified, the definitions for all of the actions for the *resource_class* are listed. Specific action names may be entered to control which actions are displayed and in what order. Actions must be separated by spaces.

resource_class
Specifies the name of the resource class with the action definitions that you want to display. If no *resource_class* parameter is specified, a list of all of the resource class names is displayed.

Security

The user needs read permission for the *resource_class* specified in **lsactdef** to run **lsactdef**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0 Command has run successfully.

1 Error occurred with RMC.

2 Error occurred with CLI script.

3 Incorrect flag on command line.

4 Incorrect parameter on command line.

5 Error occurred with RMC that was based on faulty command line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor

and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. 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.

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

1. To list the names of all of the resource classes, enter:

```
lsactdef
```

   The output will look like this:

```
class_name
"IBM.Association"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EventResponse"
"IBM.Host"
"IBM.Program"
"IBM.Sensor"
"IBM.ManagedNode"
...
```

2. To list the public resource action definitions for resource class IBM.AuditLog, enter:

```
lsactdef IBM.AuditLog
```

   The output will look like this:

```
Resource Action Definitions for
class_name: IBM.AuditLog
action 1:
        action_name    = "GetRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 0
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
action 2:
        action_name    = "DeleteRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 1
```

```
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
....
```

3. To list the structured data definition required for invoking the action on resources in resource class IBM.AuditLog, action GetRecords, enter:

```
lsactdef -s i IBM.AuditLog GetRecords
```

The output will look like this:

```
Resource Action Input for: IBM.AuditLog
action_name GetRecords:
SD element 1:
        element_index     = 0
        element_name      = "MatchCriteria"
        description       = ""
        element_data_type = "char_ptr"
        display_name      = ""
SD element 2:
        element_index     = 1
        element_name      = "IncludeDetail"
        description       = ""
        element_data_type = "uint32"
        display_name      = ""
```

Location

/usr/sbin/rsct/bin/lsactdef Contains the **lsactdef** command

Related Information

Commands: **lsrsrcdef**
Files: **rmccli**
Books: see the *RSCT Administration Guide* for information about RMC operations

---

lsallq Command

Purpose

Lists the names of all configured queues.

Syntax

lsallq [**-c**]

Description

The **lsallq** command lists the names of all configured queues contained in the **/etc/qconfig** file. By specifying the **-c** flag, this listing is displayed in colon format. This flag is used mainly by SMIT.

You can use the Printer Queues application in Web-based System Manager (wsm) to change print queue characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsallq** fast path to run this command.

Flag

-c Causes colon format output for use by SMIT.

Examples

1. To list all of the queue names in the **/etc/qconfig** file, enter:

   ```
   lsallq
   ```

   A listing similar to the following is displayed:

   ```
   lp0
   lp1
   lp2
   ```

2. To list all configured queues in colon format, enter:

   ```
   lsallq  -c
   ```

   A listing similar to the following is displayed:

   ```
   lp0
   lp0:queue1
   lp0:queue2
   lp1
   ```

Files

/usr/bin/lsallq	Contains the **lsallq** command.
/etc/qconfig	Configuration file.

Related Information

The **chque** command, **lsque** command, **lsallqdev** command, **mkque** command, **rmque** command.

The **qconfig** file.

Listing Print Queues and Print Queue Devices in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

lsallqdev Command

Purpose

Lists all configured printer and plotter queue device names within a specified queue.

Syntax

lsallqdev [**-c**] **-q***Name*

Description

The **lsallqdev** command lists all configured device names within a specified queue in the **/etc/qconfig** file.

You can use the Printer Queues application in Web-based System Manager (wsm) to change print queue characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsallqdev** fast path to run this command.

Flags

-q *Name* Specifies the queue name.

-c Causes colon format output for use by SMIT.

Examples

1. To list the names all of the configured queue devices within the `lp0` queue in the **/etc/qconfig** file, enter:

   ```
   lsallqdev  -q lp0
   ```

   A listing similar to the following is displayed:

   ```
   lpd0
   lpd1
   lpd2
   ```

2. To list the names of all of the configured queue device within the `lp0` queue in the **/etc/qconfig** file in colon format, enter:

   ```
   lsallqdev  -q lp0  -c
   ```

   A listing similar to the following is displayed:

   ```
   lp0:lpd1
   lp0:lpd2
   ```

Files

/usr/bin/lsallqdev	Contains the **lsallqdev** command.
/etc/qconfig	Configuration file.

Related Information

The **chquedev** command, **lsquedev** command, **lsallq** command, **mkquedev** command, **rmquedev** command.

The **qconfig** file.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Listing Print Queues and Print Queue Devices in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

---

lsattr Command

Purpose

Displays attribute characteristics and possible values of attributes for devices in the system.

Syntax

lsattr { **-D** [**-O**] | **-E** [**-O**] | **-F** *Format* [**-Z** *Character*] } **-l** *Name* [**-a** *Attribute*] ...[**-f** *File*] [**-h**] [**-H**]

lsattr { **-D** [**-O**] | **-F** *Format* [**-Z** *Character*] } { [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } [**-a** *Attribute*] ... [**-f** *File*] [**-h**] [**-H**]

lsattr -R { **-l** *Name* | [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } **-a** *Attribute* [**-f** *File*] [**-h**] [**-H**]

lsattr -l *Name* { **-o** *operation* [...] } **-F** *Format* [**-Z** *Character*] [**-f** *File*] [**-h**] [**-H**]

lsattr { [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] } { **-o** *operation* [...] } **-F** *Format* [**-Z** *Character*] [**-f** *File*] [**-h**] [**-H**]

Description

The **lsattr** command displays information about the attributes of a given device or kind of device. If you do not specify the device logical name (**-l** *Name*), you must use a combination of one or all of the **-c** *Class*, **-s** *Subclas*s, and **-t** *Type* flags to uniquely identify the predefined device.

You must specify one of the following flags with the **lsattr** command:

-D	Displays default values.
-E	Displays effective values (valid only for customized devices specified with the **-l** flag).
-F *Format*	Specifies the user-defined format.
-R	Displays the range of legal values.

When displaying the effective values of the attributes for a customized device, the information is obtained from the Configuration database, not the device. Generally the database values reflect how the device is configured, unless it is reconfigured with the **chdev** command using the **-P** or **-T** flag. If this has occurred, the information displayed by the **lsattr** command might not correctly indicate the current device configuration until after the next system boot.

If you use the **-D** or **-E** flag, the output defaults to the values for the attribute's name*,* value*,* description*,* and user-settable strings, unless also used with the **-O** flag. The **-O** flag displays the names of all attributes specified, separated by colons. On the next line, the **-O** flag displays all the corresponding attribute values, separated by colons. The **-H** flag can be used with either the **-D**, **-E**, or **-F** flag to display headers above the column names. You can define the format of the output with a user-specified format where the *Format* parameter is a quoted list of column names separated by nonalphanumeric characters or white space using the **-F** *Format* flag. If the **-F** *Format* flag is specified, the **-Z** *Character* flag may also be specified to change the default record separater from a `'newline'` character to the indicated *'Character'*.

The **lsattr** command can display *"operation"* information from the Extended Predefined Attribute (**PdAtXtd**) object class. This information is used by the web-based System Manager. The operation information is accessed through the **-o** *operation* flag. The **-o** *operation* flag and the **-a** *attribute* flag cannot be specified in the same invocation of the **lsattr** command. The **-o** *operation* flag is also not valid with the **-R** flag. When the **-o** *operation* flag is specified, only fields from the **PdAtXtd** object class can be specified with the **-F** *Format* flag.

You can supply the flags either on the command line or from the specified *File* parameter.

Flags

-a *Attribute*	Displays information for the specified attributes of a specific device or kind of device. You can use one **-a** flag for each attribute name or multiple attribute names. If you use one **-a** flag for multiple attribute names, the list of attribute names must be enclosed in quotes with spaces between the names. Using the **-R** flag, you must specify only one **-a** flag with only one attribute name. If you do not specify either the **-a** or **-R** flag, the **lsattr** command displays all information for all attributes of the specified device. The **-a** *Attribute* flag cannot be used in conjunction with the **-o** *Operation* flag. This combination of flags causes the **lsattr** command to exit with an error message.
-c *Class*	Specifies a device class name. This flag can be used to restrict the output to that for devices of a specified class. This flag cannot be used with the **-E** or **-l** flag.

-D Displays the attribute names, default values, descriptions, and user-settable flag values for a specific device when not used with the **-O** flag. The **-D** flag displays only the attribute name and default value in colon format when used with the **-O** flag. This flag can be used with any combination of the **-c**, **-s**, and **-t** flags that uniquely identifies a device from the Predefined Devices object class or with the **-l** flag. This flag cannot be used with the **-E**, **-F**, or **-R** flag.

-E Displays the attribute names, current values, descriptions, and user-settable flag values for a specific device when not used with the **-O** flag. The **-E** flag displays only the attribute name and current value in colon format when used with the **-O** flag. This flag cannot be used with the **-c**, **-D**, **-F**, **-R**, **-s**, or **-t** flag.

-f *File* Reads the needed flags from the *File* parameter.

-F *Format* Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names separated by nonalphanumeric characters or white space. Using white space as the separator, the **lsattr** command displays the output in aligned columns. Only column names from the Predefined Attributes (**PdAt**), Customized Attributes (**CuAt**), and the Extended Predefined Attributes (**PdAtXtd**) object classes can be specified. In addition to the column names, there are two special purpose names that can be used. The name *description* can be used to obtain a display of attribute descriptions and *user-settable* can be used to obtain an indication as to whether or not an attribute can be changed. This flag cannot be used with the **-E**, **-D**, **-O** or **-R** flag.

-H Displays headers above the column output. To use the **-H** flag with the **-O** flag is meaningless, the **-O** flag prevails. To use the **-H** flag with the **-R** flag is meaningless, the **-R** flag prevails.

-h Displays the command usage message.

-l *Name* Specifies the device logical name in the Customized Devices object class whose attribute names or values are to be displayed.

-o *Operation* Displays information for the specified operations of a specific device or kind of device. You can use one **-o** flag for each operation name or multiple operation names. If you use one **-o** flag for multiple operation names, the list of attribute names must be enclosed in quotes with spaces between the names. Wildcard characters may also be used for the operation name. The valid set of wildcard charcters are the same set used by the **odmget** command. All operations associated with a specific device, or kind of device, can be displayed by using an operation value of `"?*"`. The **-o** *Operation* flag cannot be used in conjunction with the **-a** *attribute* flag or the **-R** flag. Any combination of these flags causes the **lsattr** command to exit with an error message.

-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 **-F** and **-R** flags.

-R Displays the legal values for an attribute name. The **-R** flag cannot be used with the **-D**, **-E**, **-F** and **-O** flags, but can be used with any combination of the **-c**, **-s**, and **-t** flags that uniquely identifies a device from the Predefined Devices object class or with the **-l** flag. The **-R** flag displays the list attribute values in a vertical column as follows:

```
Value1
Value2
.
.
ValueN
```

The **-R** flag displays the range attribute values as `x...n(+i)` where `x` is the start of the range, `n` is the end of the range, and `i` is the increment.

-s *Subclass* Specifies a device subclass name. This flag can be used to restrict the output to that for devices of a specified subclass. This flag cannot be used with the **-E** or **-l** flag.

-t *Type* Specifies a device type name. This flag can be used to restrict the output to that for devices of a specified class. This flag cannot be used with the **-E** or **-l** flag.

-Z *Character* The **-Z** *Character* flag is meant to be used for programs that need to deal with ODM fields that may have embbeded new line characters. An example of such a program is the Web-based System Manager. The **-Z** *Character* flag is used to change the record separator character for each record (line) of output generated. The new record separator is the *'Character'* argument to this flag. The **-Z** *Character* flag is only relevant when the **-F** *Format* flag is specified. The **-Z** *Character* flag cannot be used with the **-D**, **-E**, **-O**, or the **-R** flags.

Examples

1. To list the current attribute values for the tape device `rmt0`, type:

   ```
   lsattr -l rmt0 -E
   ```

   The system displays a message similar to the following:

   ```
   mode          yes   Use DEVICE BUFFERS during writes     True
   block_size    1024  BLOCK size (0=variable length)       True
   extfm         yes   Use EXTENDED file marks              True
   ret           no    RETENSION on tape change or reset    True
   density_set_1 37    DENSITY setting #1                   True
   density_set_2 36    DENSITY setting #2                   True
   compress      yes   Use data COMPRESSION                 True
   size_in_mb    12000 Size in Megabytes                    False
   ret_error     no    RETURN error on tape change or reset True
   ```

2. To list the default attribute values for the tape device `rmt0`, type:

   ```
   lsattr -l rmt0 -D
   ```

   The system displays a message similar to the following:

   ```
   mode          yes   Use DEVICE BUFFERS during writes     True
   block_size    1024  BLOCK size (0=variable length)       True
   extfm         yes   Use EXTENDED file marks              True
   ret           no    RETENSION on tape change or reset    True
   density_set_1 37    DENSITY setting #1                   True
   density_set_2 36    DENSITY setting #2                   True
   compress      yes   Use data COMPRESSION                 True
   size_in_mb    12000 Size in Megabytes                    False
   ret_error     no    RETURN error on tape change or reset True
   ```

3. To list the current value of the **bus_intr_lvl** attribute for the SCSI adapter `scsi0`, type:

   ```
   lsattr -l scsi0 -a bus_intr_lvl -E
   ```

   The system displays a message similar to the following:

   ```
   bus_intr_lvl 1 Bus interrupt level False
   ```

4. To list the possible values of the login attribute for the tty device `tty0`, type:

   ```
   lsattr -l tty0 -a login -R
   ```

   The system displays a message similar to the following:

   ```
   enable
   disable
   share
   delay
   ```

5. To list the default attribute values for an IBM 4340 parallel printer, type:

   ```
   lsattr -c printer -s parallel -t ibm4340 -D
   ```

   The system displays a message similar to the following:

   ```
   ptop       600       Printer TIME OUT period                  True
   line       60        Number of LINES per page                 True
   col        80        Number of COLUMNS per page               True
   ind        0         Number of columns to INDENT              True
   plot       no        Send all characters to printer UNMODIFIED  True
   backspace  yes       Send BACKSPACES                          True
   cr         yes       Send CARRIAGE RETURNS                    True
   form       yes       Send FORM FEEDS                          True
   lf         yes       Send LINE FEEDS                          True
   addcr      yes       Add CARRIAGE RETURNS to LINE FEEDS       True
   case       no        Convert lowercase to UPPERCASE           True
   tabs       yes       EXPAND TABS on eight position boundaries   True
   ```

```
wrap       no        WRAP CHARACTERS beyond the specified width True
mode       no        Return on ERROR                            True
interface  standard  Type of PARALLEL INTERFACE                 True
autoconfig available STATE to be configured at boot time        True
busy_delay 0         Microseconds to delay between characters   True
```

6. To list the possible values of the **ptop** attribute for an IBM 4340 parallel printer, type:

   ```
   lsattr -c printer -s parallel -t ibm4340  -a ptop -R
   ```

   The system displays a message similar to the following:

   ```
   1...1000 (+1)
   ```

7. To list the current attribute values for the tape device `rmt0` in colon-separated format, type:

   ```
   lsattr -l rmt0 -E -O
   ```

   The system displays a message similar to the following:

   ```
   #mode:block_size:extfm:ret:density_set_1:density_set_2:compress:size_in_mb:ret_error
   yes:1024:yes:no:37:36:yes:12000:no
   ```

8. To display system attributes, type:

   ```
   lsattr -E -l sys0
   ```

   The system displays output similar to the following:

   ```
   keylock      normal        State of system keylock at boot time
   False
   maxbuf       20            Maximum number of pages in block I/O
   BUFFER CACHE True
   maxmbuf      0             Maximum Kbytes of real memory allowed for
   MBUFS   True
   maxuproc     128           Maximum number of PROCESSES allowed per
   user      True
   autorestart  false         Automatically REBOOT system after a crash
   True
   iostat       false         Continuously maintain DISK I/O history
   True
   realmem      4194304       Amount of usable physical memory in Kbytes
   False
   conslogin    enable        System Console Login
   False
   fwversion    IBM,SPH00221  Firmware version and revision levels
   False
   maxpout      0             HIGH water mark for pending write I/Os per
   file   True
   minpout      0             LOW water mark for pending write I/Os per
   file    True
   fullcore     false         Enable full CORE dump
   True
   pre430core   false         Use pre-430 style CORE dump
   True
   ncargs       6             ARG/ENV list size in 4K byte blocks
   True
   rtasversion  1             Open Firmware RTAS version
   False
   modelname    IBM,7044-270  Machine name
   False
   systemid     IBM,011037D1F Hardware system identifier
   False
   boottype     disk          N/A
   False
   SW_dist_intr false         Enable SW distribution of interrupts
   True
   cpuguard     disable       CPU Guard
   True
   frequency    93750000      System Bus Frequency
   False
   ```

Note: The same information is available in a more readable format using SMIT (System Environments -> Change / Show Characteristics of Operating Systems).

Files

/usr/sbin/lsattr	Contains the **lsattr** command.

Related Information

The **chdev** command, **lsconn** command, **lsdev** command, **lsparent** command, **mkdev** command, **rmdev** command.

Summary of Tunable Parameters in *AIX 5L Version 5.2 Performance Management Guide*

Devices Overview for System Management in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* provides information about adding, changing, moving, and removing devices.

---

lsaudrec Command

Purpose

Lists records from the audit log.

Syntax

lsaudrec [**–l**] [**–a** | **–n** *node_name1*[**,***node_name2*]...] [**–S** *subsystem_name*] [**–s** *selection_string*] [**–x**] [**–h**] [*field_name1* [*field_name2*...]]

Description

The **lsaudrec** command is used to list records in the audit log.

The audit log is a facility for recording information about the system's operation. It can include information about the normal operation of the system as well as failures and other errors. It is meant to augment the error log functionality by conveying the relationship of the error relative to other system activities. All detailed information about failures is still written to the AIX error log.

Records are created in the audit log by subsystems that have been instrumented to do that. For example, the event response subsystem runs in the background to monitor administrator-defined conditions and then invokes one or more actions when a condition becomes true. Because this subsystem runs in the background, it is difficult for the operator or administrator to understand the total set of events that occurred and the results of any actions that were taken in response to an event. Because the event response subsystem records its activity in the audit log, the administrator can easily view its activity as well as that of other subsystems using this command.

Each record in the audit log contains named fields. Each field contains a value that provides information about the situation corresponding to the record. For example, the field named **Time** indicates the time at which the situation occurred. Each record has a set of common fields and a set of subsystem-specific fields. The common fields are present in every record in the audit log. The subsystem-specific fields vary from record to record. Their names are only significant when used with a subsystem name because they may not be unique across all subsystems. Each record is derived from a template that defines which subsystem-specific fields are present in the record and defines a format string that is used to generate a message describing the situation. The format string may use record fields as inserts. A subsystem typically has many templates.

The field names can be used as variables in a *selection string* to choose which records are displayed. A selection string is an expression that is made up of field names, constants, and operators. The syntax of a selection string is similar to an expression in the C programming language or the SQL ″where″ clause. The selection string is matched against each record using the referenced fields of each record to perform the match. Any records that match are displayed. The selection string is specified with the **-s** flag. For information on how to specify selection strings, see *RSCT Administration Guide*.

You can also specify field names as parameters to this command to choose which fields are displayed and the order in which they are displayed. The common field names are:

Time Specifies the time when the situation occurred that the record corresponds to. The value is a 64-bit integer and represents the number of microseconds since Unix Epoch (00:00:00 GMT January 1, 1970). See the constants below for specifying the time in more user-friendly formats.

Subsystem
Specifies the subsystem that generated the record. This is a string.

Category
Indicates the importance of the situation corresponding to the audit record, as determined by the subsystem that generated the record. The valid values are: **0** (informational) and **1** (error).

SequenceNumber
Specifies the unique 64-bit integer that is assigned to the record. No other record in the audit log will have the same sequence number.

TemplateId
Specifies the subsystem-dependent identifier that is assigned to records that have the same content and format string. This value is a 32-bit unsigned integer.

NodeName
Specifies the name of the node from which the record was obtained. This field name cannot be used in a selection string.

In addition to the constants in expressions that are described in *RSCT Administration Guide*, you can use the following syntax for dates and times with this command:

#*mmddhhmmyyyy*
This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour, *mm* = minutes, *yyyy* = year. For example, **#010523042002** corresponds to January 5, 11:04 PM, 2002. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 0, hour = 0, day = 1, and month = the current month.

#-mmddhhmmyyyy
This format is similar to the previous one, but is relative to the current time and date. For example, the value **#-0001** corresponds to one day ago and the value **#-010001** corresponds to one month and one hour ago. Fields can be omitted starting from the right and are replaced by 0.

The audit records considered for display and matched against the selection string can be restricted to a specific subsystem by using the **-S** flag. If this flag is specified, the subsystem-specific field names can be used in the selection string in addition to the common field names.

The nodes from which audit log records are considered for display and matched against the selection string can be restricted to a set of specific nodes by using the **-n** flag. If this flag is specified, the search is limited to the set of nodes listed. Otherwise, the search is performed for all nodes defined within the current management scope, as determined by the CT_MANAGEMENT_SCOPE environment variable.

The audit records are displayed in a table. Field names specified as parameters control which fields are displayed and the order in which they appear on each line. By default, the columns displayed are: the date

and time, the subsystem name that generated the record, the severity of the situation, and the subsystem-specific message that describes the situation. If the management scope is not local, the node name is displayed in the first column.

Flags

-l Indicates that long output should be produced. Long output includes subsystem-specific fields that are not included in the formatted message text.

-a Specifies that records from all nodes in the domain are to be displayed. If both the **-n** and the **-a** flags are omitted, records from the local node only are displayed.

-n *node_name1***[,***node_name2***]...**
Specifies the list of nodes containing audit log records that will be examined and displayed if they meet the other criteria, such as matching the specified selection string. Node group names can also be specified, which are expanded into a list of node names. If both the **-n** and the **-a** flags are omitted, records from the local node only are displayed.

-S *subsystem_name*
Specifies a subsystem name. If this flag is present, only records identified by *subsystem_name* are considered for display. The records displayed can be further restricted by the **-s** flag. If the subsystem name contains any spaces, it must be enclosed in single or double quotation marks.

For backward compatibility, the subsystem name can be specified using the **-n** flag *only* if the **-a** and the **-S** flags are *not* specified.

-s *selection_string*
Specifies a selection string. This string is evaluated against each record in the audit log. All records that match the selection string will be displayed. If the selection string contains any spaces, it must be enclosed in single or double quotation marks. For information on how to specify selection strings, see *RSCT Administration Guide*.

The names of fields in the record can be used in the expression. If the **-S** flag is not specified, only the names of common fields can be used. See the **Description** for a list of the common field names and their data types. If the **-S** flag is specified, the name of any field for the specified subsystem as well as the common field names can be used.

If this flag is omitted, the records that are displayed will depend on the **-S** flag. If the **-S** flag is omitted, all records from the audit log are displayed. Otherwise, all records for the subsystem identified by the **-S** flag are displayed.

-x Excludes the header (suppresses header printing).

-h Writes the command's usage statement to standard output.

Parameters

field_name1 **[***field_name2***...]**
Specifies one or more fields in the audit log records to be displayed. The order of the field names on the command line corresponds to the order in which they are displayed. If no field names are specified, **Time**, **Subsystem**, **Severity**, and **Message** are displayed by default. If the management scope is not local, **NodeName** is displayed as the first column by default. See the **Description** for information about these and other fields.

Security

In order to list records from an audit log when the **-S** flag is omitted, a user must have read access to the target resource class on each node from which records are to be listed. When the **-S** flag is specified, the user must have read access to the audit log resource corresponding to the subsystem identified by the **-S** flag on each node from which records are to be listed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

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 is established. 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 can be affected by this command.

CT_MANAGEMENT_SCOPE

Determines (in conjunction with the **-a** and **-n** flags) the management scope that is used for the session with the RMC daemon. The management scope determines the set of possible target nodes where audit log records can be listed. If the **-a** and **-n** flags are not specified, local scope is used. When either of these flags is specified, CT_MANAGEMENT_SCOPE is used to determine the management scope directly. 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.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Examples

1. To list all records in the audit log on every node in the current management scope as determined by the CT_MANAGMENT_SCOPE environment variable, enter:

   ```
   lsaudrec
   ```

2. To list all records that were logged in the last hour on every node in the current management scope as determined by the CT_MANAGMENT_SCOPE environment variable, enter:

   ```
   lsaudrec -s "Time > #-000001"
   ```

3. To list the time and sequence number of every record in the audit log for the subsystem **abc** on nodes **mynode** and **yournode**, enter:

   ```
   lsaudrec -n mynode,yournode -S abc Time SequenceNumber
   ```

Location

/usr/sbin/rsct/bin/lsaudrec Contains the **lsaudrec** command

Related Information

Commands: **rmaudrec**

---

lsauthent Command

Purpose

Lists the authentication methods currently configured on the system.

Syntax

lsauthent

Description

The **lsauthent** command calls the **get_auth_method** subroutine in the **libauthm.a** library, translates a list of authentication methods returned, and prints the authentication methods configured to **stdout**. Each authentication method is outputted on a separate line.

The authentication methods are listed in the order in which they are configured. If none of the authentication methods are configured, **lsauthent** returns without printing anything.

The **lsauthent** command writes an error message to **stderr** and returns a -1 if **get_auth_method** fails.

Examples

If all of the authentication methods are configured as:

```
lsauthent
```

the output would consist of:

```
Kerberos 5
Kerberos 4
Standard AIX
```

Related Information

The **chauthent** command, **ftp** command, **rcp** command, **rlogin** command, **rsh** command, **telnet**, **tn**, or **tn3270** command.

The **get_auth_method** and **set_auth_method** subroutines.

Network Overview in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Secure Rcmds in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

lsC2admin Command

Purpose

Display the name of the current C2 System Administrative Host.

Syntax

lsC2admin

Description

The **lsC2admin** command displays the name of the administrative host. An administrative host must have been defined, and the system must have been installed in C2 mode for this command to operate successfully.

Exit Status

0 The administrative host information has been successfully displayed.

1 This system was not installed with C2 security.

2 This system has not been initialized to operate in C2 mode.

3 An error occurred while displaying the name of the administrative host.

Files

/usr/sbin/lsC2admin	Contains the lsC2admin command.

Related Information

The **chC2admin** command, **isC2host** command, **mkC2admin** command, **rmC2admin** command.

---

lsCCadmin Command

Purpose

Display the name of the current Common Criteria enabled System Administrative Host.

Syntax

lsCCadmin

Description

The **lsCCadmin** command displays the name of the administrative host. An administrative host must have been defined, and the system must have been installed in Common Criteria enabled mode for this command to operate successfully.

Exit Status

0 The administrative host information has been successfully displayed.

1 This system was not installed with Common Criteria enabled security.

2 This system has not been initialized to operate in Common Criteria enabled mode.

3 An error occurred while displaying the name of the administrative host.

Files

/usr/sbin/lsCCadmin	Contains the lsCCadmin command.

Related Information

The **chCCadmin** command, **isCChost** command, **mkCCadmin** command, **rmCCadmin** command.

lscfg Command

Purpose

Displays configuration, diagnostic, and vital product data (VPD) information about the system.

Syntax

To Display Specific Data on all Systems

lscfg [**-v**] [**-p**] [**-s**] [**-l** *Name*]

Description

If you run the **lscfg** command without any flags, it displays the name, location and description of each device found in the current Customized VPD object class that is a child device of the **sys0** object. It will not display any device that has been marked `missing` in the Customized Device Object Class. The list is sorted by parent, child, and device location. Information on a specific device can be displayed with the **-l** flag.

Use the **lscfg** command to display vital product data (VPD) such as part numbers, serial numbers, and engineering change levels from either the Customized VPD object class or platform specific areas. Not all devices contain VPD data.

VPD data that is preceded by `ME` signifies that the VPD data was entered manually using a diagnostic service aid. For some devices, the vital product data is collected automatically from the devices through methods and added to the Customized VPD object class.

If you run the **lscfg** command with the **-p** flag, it displays device information stored in the platform specific data areas. When used with the **-v** flag, VPD data stored for these devices is also displayed. This information is obtained on a Common Hardware Reference Platform (CHRP) system from the open firmware device tree.

-l *Name*	Displays device information for the named device.
-p	Displays the platform-specific device information. This flag only applies to AIX 4.2.1 or later.
-v	Displays the VPD found in the Customized VPD object class. Also, on AIX 4.2.1 or later, displays platform specific VPD when used with the **-p** flag.
-s	Displays the device description on a separate line from the name and location.

Examples

1. To display the system configuration, enter:

   ```
   lscfg
   ```

   The system displays a message similar to the following:

   ```
   INSTALLED RESOURCE LIST
   ```

   The following resources are installed on the machine:

   ```
   +/- = Added or deleted from Resource List.
   *   = Diagnostic support not available.

     Model Architecture: chrp
     Model Implementation: Multiple Processor, PCI bus

   + sys0                              System Object
   + sysplanar0                        System Planar
   + mem0                              Memory
   + L2cache0                          L2 Cache
   ```

```
+ proc0            U1.1-P1-C1      Processor
* pci3             U0.2-P1         PCI Bus
+ scsi0            U0.1-P1/Z1      Wide/Ultra-2 SCSI I/O Controller
+ rmt0             U1.1-P1/Z1-A3   SCSI 4mm Tape Drive (12000 MB)
+ cd0              U1.1-P1/Z1-A5   SCSI Multimedia CD-ROM Drive (650 MB)
+ hdisk0           U1.1-P1/Z1-A9   16 Bit LVD SCSI Disk Drive (4500 MB)
+ fd0              U0.1-P1-D1      Diskette Drive
..
..
```

2. To display the system configuration with the device description on a separate line, enter:

```
lscfg  -s
```

   The system displays a message similar to the following:

```
INSTALLED RESOURCE LIST
```

   The following resources are installed on the machine:

```
+/- = Added or deleted from Resource List.
*   = Diagnostic support not available.

  Model Architecture: chrp
  Model Implementation: Multiple Processor, PCI bus

+ sys0             U1.1-P1-C2
        System Object
+ sysplanar0
        System Planar
+ mem0
        Memory
+ L2cache0
        L2 Cache
+ proc0            U1.1-P1-C1
        Processor
+ proc1            U1.1-P1-C1
        Processor
+hdisk0   U1.1-P1/Z1-A9
      16 Bit LVD SCSI Disk Drive (4500 MB)
+fd0      U1.1-P1-D1
      Diskette Drive

..
..
```

3. To display the name, location, and description for devices specified by the logical name proc without VPD, enter:

```
lscfg -lproc\*
```

   The system displays information for all devices with logical names beginning with proc, as follows:

```
  proc0            U1.1-P1-C1  Processor
  proc1            U1.1-P1-C1  Processor
  proc2            U1.1-P1-C1  Processor
  proc3            U1.1-P1-C1  Processor
  proc4            U1.1-P1-C2  Processor
  proc5            U1.1-P1-C2  Processor
  proc6            U1.1-P1-C2  Processor
  proc7            U1.1-P1-C2  Processor
```

4. To display the VPD for a specific device specified by the logical name ent0, enter:

```
lscfg  -v -l ent0
```

   The system displays the following:

```
 ent0             U0.1-P1-I2/E1   Gigabit Ethernet-SX PCI Adapter (14100401)

        Network Address.............0004AC7C00C4
```

```
        Displayable Message.........Gigabit Ethernet-SX PCI Adapter (14100401)

        EC Level....................E77998
        Part Number.................07L8916
        FRU Number..................07L8918
        Device Specific.(YL)........U0.1-P1-I2/E1
```

5. To display the VPD in the open firmware device tree for the corresponding node to the ent0 device, enter:

```
lscfg -vp -lent0
```

   The following is displayed:

```
  ent0             U0.1-P1-I2/E1  Gigabit Ethernet-SX PCI Adapter (14100401)

        Network Address.............0004AC7C00C4
        Displayable Message.........Gigabit Ethernet-SX PCI Adapter (14100401)

        EC Level....................E77998
        Part Number.................07L8916
        FRU Number..................07L8918
        Device Specific.(YL)........U0.1-P1-I2/E1


  PLATFORM SPECIFIC

  Name:  ethernet
    Model:  Galaxy, EtherLink 1000-SX-IBM
    Node:  ethernet@1
    Device Type:  network
    Physical Location: U0.1-P1-I2/E1
```

Files

/usr/sbin/lscfg	Contains the **lscfg** command.

Related Information

The **lsattr** command, **lsconn** command, **lsdev** command, **lsparent** command.

---

lsclass Command

Purpose

List Workload Management classes and their limits.

Syntax

lsclass [**-C** | **-D** | **-f**] [**-r**] [**-d** *Config_Dir*] [**-S** *SuperClass*] [*Class*]

Description

The **lsclass** command, with no argument, returns the list of superclasses, one per line. With a class name as argument, it prints the class. The subclasses can be displayed with the **-r** (recursive) flag, or with the **-S** *Superclass* flag.

When WLM is started, if an empty string is passed as the name of the configuration with the **-d** flag, **lsclass** lists the classes defined in the in-core WLM data structures.

The **lsclass** command does not require any special level of privilege and is accessible for all users.

Note: If this command is given a set of time-based configurations (either specified with the **-d** flag, or because the current configuration is a set), the **lsclass** command returns the classes of the regular configuration which applies (or would apply) at the time the command is issued.

Flags

-C Displays the class attributes and limits in colon-separated records, as follows:

```
lsclass -C myclass
#name:description:tier:inheritance:authuser:authgroup:adminuser:admingroup:rset:CPUshares:CPUmin:
myclass::0:no:root:system:root:system::-:0:50:100:4:10:50:75:6:0:100:100:-:s:-:KB:-:s:-:-:-:no
```

-d *Config_Dir* Use **/etc/wlm/***Config_Dir* as alternate directory for the definition files. If an empty string is passed (for example, `-d ″″`), **lsclass** lists the classes defined in the in-core WLM data structures. If this flag is not present, the current configuration files in the directory pointed to by **/etc/wlm/current** are used.

-D Displays the default values for the class attributes and limits in colon-separated records. Any other flag or argument used in conjunction with **-D** is ignored. For example:

```
lsclass -D
#name:description:tier:inheritance:authuser:authgroup:adminuser:admingroup:rset:CPUshares:CPUmin:
::0:no::::::-:0:100:100:-:0:100:100:-:0:100:100:-:s:-:KB:-:s:-:-:-:no
```

-f Displays the output in stanzas, with each stanza identified by a class name. Each *Attribute*=*Value* pair is listed on a separate line:

```
Class:
        attribute1=value
        attribute2=value
        attribute3=value
```

-r Displays, recursively, the superclasses with all their subclasses. When specifying **-r**:

- If *Class* is not specified, **lsclass** shows all the superclasses with all their subclasses.
- If the name of a superclass is specified, **lsclass** displays the superclass with all its subclasses.
- If the name of a subclass is specified, **-r** is ineffective (displays only the subclass).

-S *SuperClass* Restricts the scope of the command to the subclasses of the specified superclass. Only subclasses are shown with the **-S** flag.

Files

classes	Contains the names and definitions of the classes.
limits	Contains the limits enforced on the classes.
shares	Contains the resource shares attributes for each class.

Related Information

The **wlmcntrl** command, **chclass** command, **mkclass** command, and **rmclass** command.

---

lscomg Command

Purpose

Displays information about the communication groups of a peer domain.

Syntax

lscomg [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–i**] [**–h**] [**–TV**] [*communication_group*]

Description

The **lscomg** command displays information about the communication groups that are defined to the online peer domain on which the command runs. If you specify the name of a communication group, the **lscomg** command displays information about that communication group only.

Some of the communication group information that is displayed follows:

Name Specifies the name of the communication group.

Sensitivity Specifies the number of missed heartbeats that constitute a failure.

Period Specifies the number of seconds between heartbeats.

Priority Specifies the relative priority of the communication group.

Broadcast Indicates whether broadcast should be used if it is supported by the underlying media.

SourceRouting
Indicates whether source routing should be used if it is supported by the underlying media.

NIMPath Specifies the path to the Network Interface Module (NIM) that supports the adapter types in the communication group.

NIMParameters
Specifies the NIM start parameters.

Use the **-i** flag to display information about the interface resources that refer to *communication_group*. If you specify the **-i** flag, **lscomg** displays the following information:

Name Specifies the name of the interface resource that refers to *communication_group*.

NodeName Specifies the host name of the interface resource that refers to *communication_group*.

IPAddr Specifies the IP address of the interface resource that refers to *communication_group*.

SubnetMask Specifies the subnet mask of the interface resource that refers to *communication_group*.

Subnet Specifies the subnet of the interface resource that refers to *communication_group*.

Flags

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default format.

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x Excludes the header (suppresses header printing).

-i Displays information about the interface resource that refers to *communication_group*.

-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

communication_group
Specifies one or more communication groups to display information about. You can specify a communication group name or a substring of a communication group name for this parameter. If you specify a substring, the command displays information about any defined communication group with a name that contains the substring.

Security

The user of the **lscomg** command needs read permission for the **IBM.CommunicationGroup** resource class. Read permission for the **IBM.NetworkInterface** resource class is required to display the network interface information. By default, **root** on any node in the peer domain has read and write access to these resource classes 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 communication group 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 defined and online to the peer domain on which the communcation group exists.

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 defined and online to peer domain **ApplDomain**.

1. To display general information about the communication groups for **ApplDomain**, run this command on **nodeA**:

```
lscomg
```

   The output will look like this:

```
Name  Sensitivity  Period  Priority  Broadcast  SourceRouting  NIMPath                      NIMParameters
ComG1      2          2        1        no          yes        /usr/sbin/rsct/bin/hats_nim      -l 5
```

2. To display information about the interface resources that refer to the communication group **ComGrp1** for the peer domain **ApplDomain**, run this command on **nodeA**:

```
lscomg -i ComGrp1
```

   The output will look like this:

```
Name    NodeName    IPAddr         SubnetMask       Subnet
eth0    n24         9.234.32.45    255.255.255.2    9.235.345.34
eth0    n25         9.234.32.46    255.255.255.2    9.235.345.34
```

Location

/usr/sbin/rsct/bin/lscomg Contains the **lscomg** command

Related Information

Commands: **chcomg**, **lsrpnode**, **lsrpdomain**, **mkcomg**, **preprpnode**, **rmcomg**

---

lscondition Command

Purpose

Lists information about one or more conditions.

Syntax

lscondition [**–m** | **–n** | **–e**] [**–C** | **–l** | **–t** | **–d** | **–D** *delimiter*] [**–A**] [**–q**] [**–x**] [**–a**] [**–h**] [**–TV**] [*condition1* [**,***condition2***,**...]**:***node_name*]]

Description

The **lscondition** command lists the following information about defined conditions:

Name The name of the condition.

Location The location of the condition (for management domain scope or peer domain scope).

MonitorStatus The status of the condition.

ResourceClass The resource class monitored by this condition.

EventExpression The expression used in monitoring this condition.

EventDescription The text description of **EventExpression**.

RearmExpression The expression used in determining when monitoring should restart for this condition after an event has occurred.

RearmDescription The text description of **RearmExpression**.

SelectionString The selection string that is applied to the attributes of **ResourceClass** to determine which resources are included in the monitoring of this condition.

Severity The severity of the condition: critical, warning, or informational.

NodeNamelist The host names of the nodes where the condition is registered.

MgtScope The RMC scope in which the condition is monitored.

For a list of all conditions, enter the **lscondition** command without any condition names specified. A list of all the condition names is returned with the monitoring status for each condition. The default format in this case is tabular. Specifying a node name following the condition names limits the display to the conditions defined on that node. You can list all of the conditions on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope, which is determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the conditions are listed. For local scope, only conditions on the local node are listed. Otherwise, the conditions from all nodes within the domain are listed.

For all of the information about all condition names, specify the **-A** flag with the **lscondition** command. The **-A** flag causes all information about a condition to be listed when no condition names are specified. When all the information about all conditions is listed, the default format is long. If a monitoring-status flag (**-e**, **-m**, or **-n**) is specified, the conditions with that status are listed. The error information is only displayed when the **-e** flag is specified.

When more than one **condition** is specified, the condition information is listed in the order in which the condition names are entered.

By default, when a condition name is specified with the **lscondition** command, all of the condition's attributes are displayed.

Flags

–a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lscondition -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–m Lists only those conditions that are being monitored without error.

–n Lists only those conditions that are not being monitored.

–e Lists only those conditions that are monitored in error.

–C Displays the **mkcondition** command that could be used to create the condition. If more than one condition is specified, each **mkcondition** command appears on a separate line. This flag is ignored when no conditions are specified. This flag overrides the **–l** flag.

–l Produces long-formatted output. Displays the condition information on separate lines.

–t Displays the condition information in separate columns (table format).

–d Produces delimiter-formatted output. The default delimiter is a colon (**:**). Use the **–D** flag if you want to change the default delimiter.

–D *delimiter*
Produces delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–A Displays all of the attributes of the condition.

–q Does not return an error when the condition does not exist.

–x Suppresses header printing.

–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

condition1 **[**,*condition2*,**...]**
Specifies the name of an existing condition that is defined on the host name *node_name*. You can specify more than one condition name. This parameter can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring will be listed.

node_name Specifies the node 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.

Security

The user needs read permission for the **IBM.Condition** resource class to run **lscondition**. 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.

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 list all conditions and their monitoring status, run this command:

   ```
   lscondition
   ```

   The output will look like this:

   ```
   Name                      Location      MonitorStatus
   "FileSystem space used"   "nodeA"       "Monitored"
   "tmp space used"          "nodeA"       "Not monitored"
   "var space used"          "nodeA"       "Error"
   ```

2. To list general information about the condition ″FileSystem space used″ in long form, run this command:

   ```
   lscondition "FileSystem space used"
   ```

   The output will look like this:

   ```
   Name             = "FileSystem space used"
   Location         = "nodeA"
   MonitorStatus    = "Monitored"
   ResourceClass    = "IBM.FileSystem"
   EventExpression  = "PercentTotUsed > 99"
   EventDescription = "Generate event when space used is
                       greater than 99 percent full"
   RearmExpression  = "PercentTotUsed < 85"
   RearmDescription = "Start monitoring again after it is
                       less than 85 percent"
   SelectionString  = ""
   Severity         = "w"
   NodeNamelist     = "local"
   MgtScope         = "local"
   ```

3. To list the command that would create the condition ″FileSystem space used″, run this command:

   ```
   lscondition -C "FileSystem space used"
   ```

   The output will look like this:

   ```
   mkcondition -r IBM.FileSystem -a PercentTotUsed \
   -e "PercentTotUsed > 99" -E "PercentTotUsed < 85" \
   -d "Generate event when space used is greater than 99 percent full" \
   -D "Start monitoring after it is less than 85 percent" \
   -S w "FileSystem space used"
   ```

4. To list all conditions that have the string **space** in their names, run this command:

   ```
   lscondition space
   ```

   The output will look like this:

   ```
   Name             = "FileSystem space used"
   MonitorStatus    = "Monitored"
   ⋮
   Name             = "tmp space used"
   MonitorStatus    = "Not Monitored"
   ```

```
⋮
Name             = "var space used"
MonitorStatus    = "Monitored"
⋮
```

5. To list the conditions that are in error and display the error information, run this command:

   ```
   lscondition -e
   ```

   The output will look like this:

   ```
   Name               MonitorStatus
   "var space used"   "Error"
   ```

This example applies to clustered systems:

1. To list all conditions and their monitoring status, run this command:

   ```
   lscondition
   ```

   The output will look like this:

   ```
   Name                       Location    MonitorStatus
   "FileSystem space used"    "nodeA"     "Monitored"
   "tmp space used"           "nodeB"     "Not monitored"
   "var space used"           "nodeC"     "Error"
   ```

Location

/usr/sbin/rsct/bin/lscondition Contains the **lscondition** command

Related Information

ERRM commands: **chcondition**, **lscondresp**, **mkcondition**, **rmcondition**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations

---

lscondresp Command

Purpose

Lists information about a condition and its linked responses, if any.

Syntax

lscondresp [**–a** | **–n**] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–q**] [**–x**] [**–z**] [**–h**] [**–TV**] [*condition*[**:***node_name*] [*response1* [*response2*...]]]

lscondresp [**–a** | **–n**] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–q**] [**–x**] [**–z**] **–r** [**–h**] [**–TV**] *response1*[**:***node_name*] [*response2*...]

Description

The **lscondresp** command lists information about a condition and its linked responses. The information shows which responses are linked with a condition and whether monitoring is active for a condition and its linked response.

The following information is listed:

Condition The name of the condition linked with a response.

Response The name of the response linked with the condition.

State The state of the response for the condition. The state indicates whether a specified response is active or not.

Location The location of the condition and the response.

To list a particular condition and response, specify both the condition and the response. To list all responses to a condition, specify the condition only. To list all conditions to which a response is linked, specify the response and the **-r** flag. To list all conditions and their linked responses, do not specify any condition or response parameters.

Specifying a node name limits the display to the linked conditions and responses defined on that node. List all of the linked conditions and responses on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the linked conditions and responses are listed. For local scope, only linked conditions and responses on the local node are listed. For management domain scope and peer domain scope, the linked conditions and responses from all nodes within the domain are listed.

When neither the **–a** flag nor the **–n** flag is specified, all selected conditions for the responses are listed. Tabular format is the default.

Flags

–a Lists only those responses that are active for the condition.

–n Lists only those responses that are not active for the condition.

–l Displays the condition information and response information on separate lines (long format).

–t Displays the condition information and response information in separate columns (table format).

–d Specifies delimiter-formatted output. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses *delimiter*. Use this flag to specify something other than the default colon (**:**). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

–q Does not return an error if either the **condition** or the *response* does not exist.

–x Suppresses header printing.

–z Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lscondresp -z** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–r Indicates that all command parameters are responses. There are no conditions specified. This lists all linked condition-response information for the specified responses.

–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

condition The *condition* can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring and is linked to the response will be listed.

response1 **[***response2***...]**
This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring and is linked to the condition will be listed.

node_name Specifies the node 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.

Security

The user needs read permission for the **IBM.Association** resource class to run **lscondresp**. 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.

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 list all conditions with their linked responses, run this command:

   ```
   lscondresp
   ```

   The output will look like this:

   ```
   Condition                 Response                      State         Location
   "FileSystem space used"   "Broadcast event on-shift"    "Active"      nodeA
   "FileSystem space used    "E-mail root any time"        "Not Active"  nodeA
   "Page in Rate"            "Log event any time"          "Active"      nodeA
   ```

2. To list information about the condition ″FileSystem space used″, run this command:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                 Response                      State         Location
   "FileSystem space used"   "Broadcast event on-shift"    "Active"      nodeA
   "FileSystem space used    "E-mail root any time"        "Not Active"  nodeA
   ```

3. To list information about the condition ″FileSystem space used″ for responses that are active, run this command:

   ```
   lscondresp -a  "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                 Response                      State      Location
   "FileSystem space used"   "Broadcast event on-shift"    "Active"   nodeA
   ```

4. To list information about the condition ″FileSystem space used″ with the linked response ″Broadcast event on-shift″, run this command:

   ```
   lscondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

   The output will look like this:

   ```
   Condition                 Response                      State      Location
   "FileSystem space used"   "Broadcast event on-shift"    "Active"   nodeA
   ```

5. To list all conditions that have the string **space** in their names with their linked responses, run this command:

   ```
   lscondresp space
   ```

   The output will look like this:

   ```
   Condition                 Response                      State          Location
   "FileSystem space used"   "Broadcast event on-shift"    "Active"       nodeA
   "FileSystem space used    "E-mail root any time"        "Not Active"   nodeA
   ```

These examples apply to management domains:

1. In this example, the condition ″FileSystem space used″ is defined on the management server. To list information about ″FileSystem space used″, run this command on the management server:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                 Response                     State         Location
   "FileSystem space used"   "Broadcast event on-shift"   "Active"      nodeB
   "FileSystem space used"   "E-mail root any time"       "Not Active"  nodeB
   ```

2. In this example, the condition ″FileSystem space used″ is defined on the managed node **nodeC**. To list information about ″FileSystem space used″, run this command on the management server:

   ```
   lscondresp "FileSystem space used":nodeC
   ```

   The output will look like this:

   ```
   Condition                 Response                     State          Location
   "FileSystem space used"   "Broadcast event on-shift"   "Active"       nodeC
   "FileSystem space used"   "E-mail root any time"       "Not Active"   nodeC
   ```

This example applies to a peer domain:

1. In this example, the condition ″FileSystem space used″ is defined in the domain. To list information about ″FileSystem space used″, run this command on one of the nodes in the domain:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                Response                     State          Location
   "FileSystem space used"  "Broadcast event on-shift"   "Active"       nodeD
   "FileSystem space used"  "E-mail root any time"       "Not Active"   nodeD
   "FileSystem space used"  "Broadcast event on-shift"   "Active"       nodeE
   "FileSystem space used"  "E-mail root any time"       "Not Active"   nodeE
   ```

Location

/usr/sbin/rsct/bin/lscondresp Contains the **lscondresp** command

Related Information

ERRM commands: **mkcondition**, **mkcondresp**, **mkresponse**, **rmcondresp**, **startcondresp**, **stopcondresp**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations

---

lsconn Command

Purpose

Displays the connections a given device, or kind of device, can accept.

Syntax

lsconn { **-p** *ParentName* | [**-c** *ParentClass*] [**-s** *ParentSubclass*] [**-t** *ParentType*] } { **-l** *ChildName* | **-k** *ChildConnectionKey* } [**-f** *File*] [**-F** *Format*] [**-h**] [**-H**]

Description

The **lsconn** command, when used with the **-p** *ParentName* flag, displays the connection locations on the parent device to which the device specified by the **-l** *ChildName* flag can be connected, or to which devices of the connection type specified by the **-k** *ChildConnectionKey* flag can be connected. If the **-k** and **-l** flags are not used, the **lsconn** command displays information as to where a child device can be connected on the specified parent.

If the **-p** *ParentName* flag is not used, you must use a combination of one or all of the **-c** *ParentClass*, **-s** *ParentSubclass*, and **-t** *ParentType* flags to uniquely identify the predefined parent device.

You can display the default output, which is the connection location (or connection location and connection key if no child is specified), from the Predefined Connection object class. If you do not display the default, you can display the output in a user-specified format where the *Format* parameter is a quoted list of column names separated by nonalphanumeric characters or white space using the **-F** *Format* flag. You can insert headers above the columns using the **-H** flag.

You can supply the flags either on the command line or from the specified *File* parameter.

Flags

-c *ParentClass*	Specifies the class name of a possible parent device in the Predefined Devices object class. This flag cannot be used with the **-p** flag.
-f *File*	Reads the needed flags from the *File* parameter.
-F *Format*	Formats the output in a user-specified format where the *Format* parameter is a quoted list of column names from the Predefined Connection object class separated and possibly terminated by nonalphanumeric characters or white space. Using white space as the separator, the **lsconn** command displays the output in aligned columns.
-H	Displays headers above the column output.
-h	Displays the command usage message.
-k *ChildConnectionKey*	Specifies the connection key that identifies the subclass of the child device. This flag cannot be used with the **-l** flag.
-l *ChildName*	Specifies the logical name of a possible child device. This flag cannot be used with the **-k** flag.
-p *ParentName*	Specifies the parent device's logical name from the Customized Devices object class. This flag cannot be used with the **-c**, **-s**, or **-t** flag.
-s *ParentSubclass*	Specifies the subclass of a possible parent device in the Predefined Devices object class. This flag cannot be used with the **-p** flag.
-t *ParentType*	Specifies the device type of a possible parent device from the Predefined Devices object class. This flag cannot be used with the **-p** flag.

Examples

1. To list all possible connection locations on the IBM 8-Port EIA-232/RS-422A (PCI) Adapter sa2 that will accept an RS-232 device connection, type:

   ```
   lsconn -p sa2 -k rs232
   ```

   The system displays a message similar to the following:

   ```
   0
   1
   2
   3
   4
   5
   6
   7
   ```

2. To list all possible connection locations and connection types on the IBM 8-Port EIA-232/RS-422A (PCI) Adapter sa2, enter:

```
lsconn -p sa2
```

The system displays a message similar to the following:

```
0 rs232
1 rs232
2 rs232
3 rs232
4 rs232
5 rs232
6 rs232
7 rs232
0 rs422
1 rs422
2 rs422
3 rs422
4 rs422
5 rs422
6 rs422
7 rs422
```

Files

/usr/sbin/lsconn	Specifies the command file.

Related Information

The **chdev** command, **lsattr** command, **lsdev** command, **lsparent** command, **mkdev** command, **rmdev** command.

Devices Overview for System Management in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* provides information about adding, changing, moving, and removing devices.

---

lscons Command

Purpose

Writes the name of the current console device to standard output.

Syntax

lscons [**-s**] [**-a** | **-O**]

lscons -b [**-s**] [**-a** | **-O**]

lscons -d [**-s**]

Description

The **lscons** command writes the name of the current console device to standard output. This command is also used to write the name of the device that is to be the console on the next start of the system to standard output. You can change the current console device using the **swcons** command. You can change the device to be the system console on the next start of the system using the **chcons** command.

Flags

-a Displays a list of *attribute name = attribute value* pairs for the console device and console logging and tagging attributes. When used with the **-b** flag, the values are retrieved from the ODM. Without the **-b** flag, the values are retrieved from the console device driver. For additional information about console output logging and tagging, see the console Special File in the *AIX 5L Version 5.2 Files Reference* book.
Note: This flag is not valid with the **-O** flag or the **-d** flag.

-b Displays the full path name of the system console selected for the next startup of the system.

-d Displays the full path name of the system console selected on the current startup of the system.
Note: This flag is not valid with the **-O** flag or the **-a** flag.

-O Similar to the **-a** flag but outputs the attribute names and values in a format suitable for use by SMIT. This flag is NOT valid with the **-d** flag.
Note: This flag is not valid with the **-d** flag or the **-a** flag.

-s Suppresses reporting of the path name.

Exit Status

This command returns the following exit values:

0 The device you are using is the current system console.

1 The device you are using is not the current system console.

2 The device you are using is the console device selected at system start but is not currently the device supporting console message output.

3 Flags specified are not valid.

4 System error occurred.

Examples

1. To display the full path name of the current system console, type:
   ```
   lscons
   ```
2. To display the full path name of the system console effective on the next startup of the system, type:
   ```
   lscons  -b
   ```
3. To display the full path name of the system console selected on the current startup of the system, type:
   ```
   lscons -d
   ```
4. To test whether or not the current system console is directed to your display, type:
   ```
   if lscons -s
   then
   echo "System messages are directed to my display" >/dev/tty
   fi
   ```

Files

/usr/sbin/lscons Contains the **lscons** command.

Related Information

The **chcons** command, **swcons** command.

The **console** special file.

---

lsdev Command

Purpose

Displays devices in the system and their characteristics.

Syntax

lsdev [**-C**][**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] [**-f** *File*] [**-F** *Format* / **-r** *ColumnName*] [**-h**] [**-H**] [**-l** { *Name* | - }] [**-p** *Parent*] [**-S** *State*]

lsdev -P [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] [**-f** *File*] [**-F** *Format* | **-r** *ColumnName*] [**-h**] [**-H**]

Description

The **lsdev** command displays information about devices in the Device Configuration database. You can display information about all devices in the Customized Devices object class using the **-C** flag. Any combination of the **-c** *Class*, **-s** *Subclass*, **-t** *Type*, **-l** *Name*, **-p** *Parent*, and **-S** *State* flags selects a subset of the customized devices. You can display information about all devices in the Predefined Devices object class using the **-P** flag. Any combination of the **-c** *Class*, **-s** *Subclass*, and **-t** *Type* flags selects a subset of the predefined devices.

You can display the default output one of two ways. You can either display the default output from the Customized Devices object class using the **-C** flag, or display the default output from the Predefined Devices object class using the **-P** flag. To override these two default outputs, you can use the **-F** *Format* flag. The **-F** *Format* flag displays the output in a user-specified format where the *Format* parameter is a quoted list of column names separated and possibly ended by nonalphanumeric characters or white space.

The **-r** *ColumnName* flag displays the range of values for a particular column over the specified set of devices.

Note that the **lsdev** command only shows information about devices based upon information in the Customized Devices (**Cudv**) object class or the Predefined Devices (**PdDv**) object class. Other object classes (such as the Customized Path (**CuPath**) object class) are not examined. This situation means that there may be conditions where a device may not be displayed. For example, if the **-p** *Parent* flag is used, but the parent identified in the Customized Devices object for a device does not match the *Parent* specified through the **-p** flag, the device will not be displayed. However, the device may have a path to the given *Parent* that is defined in the Customized Paths object class. The **lspath** command should be used to show all MPIO capable child devices of the given parent.

You can use the Devices application in Web-based System Manager to change device characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsdev** fast path to run this command.

Flags

-C	Lists information about a device that is in the Customized Devices object class. The default information displayed is *name*, *status*, *location*, and *description*. The **-C** flag is not required. It is maintained for compatibility reasons. The **-C** flag cannot be specified with the **-P** flag. If neither are specified, the **lsdev** command behaves as if the **-C** flag were specified.
-c *Class*	Specifies a device class name. This flag can be used to restrict output to devices in a specified class.
-f *File*	Reads the needed flags from the *File* parameter.

-F *Format* Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names from the Predefined or Customized Devices object class, separated and possibly ended by nonalphanumeric characters or white space. Using white space as the separator, the **lsdev** command displays the output in aligned columns.

If you specify the **-F** *Format* flag with the **-C** flag, you can specify column names from both the Customized and Predefined Devices object classes. If you specify the **-F** *Format* flag with the **-P** flag, you can only specify column names from the Predefined Devices object class. In addition to the column names, the special purpose name *description* can be used to obtain a display of device descriptions. This flag cannot be used with the **-r** *ColumnName* flag. Also, the *physloc* special purpose name can be used to display a device's physical location code.

-H Displays headers above the column output.

-h Displays the command usage message.

-l *Name* Specifies the device logical name from the Customized Devices object class for the device for which information is listed. The *Name* argument to the **-l** flag may contain the same wildcard characters that can be used with the **odmget** command. If the *Name* argument is a dash, names will be read from STDIN. Names on STDIN must be separated by a comma, a tab, a space, or a ″newline″ character. The names cannot contain wildcard characters. This flag cannot be used with the **-P** flag.

-p *Parent* Specifies the device logical name from the Customized Devices object class for the parent of devices to be displayed. The **-p** *Parent* flag can be used to show the child devices of the given *Parent*. The *Parent* argument to the **-p** flag may contain the same wildcard charcters that can be used with the **odmget** command. This flag cannot be used with the **-P** flag.

-P Lists information about a device that is in the Predefined Devices object class. The default information displayed is *class*, *type*, *subclass*, *description*. This flag cannot be used with the **-C**, **-l**, or **-S** flags.

-r *ColumnName* Displays the set of values in a column. For example, the *ColumnName* parameter takes the value of the *Class* parameter to list all the classes. If you specify the **-r** *ColumnName* flag with the **-C** flag, you can specify column names from both the Customized and Predefined Devices object classes. If you specify the **-r** *ColumnName* flag with the **-P** flag, you can only specify column names from the Predefined Devices object class. This flag cannot be used with the **-F** *Format* flag.

-S *State* Lists all devices in a specified state as named by the *State* parameter. This flag cannot be used with the **-P** flag. The *State* parameter can either be a value of *d*, *D*, *0* or *defined* for the Defined state; *a*, *A*, *1*, or *available* for the Available state; or *s*, *S*, *2*, or *stopped* for the Stopped state. This flag can be used to restrict output to devices in a specified state.

-s *Subclass* Specifies a device subclass name. This flag can be used to restrict output to devices in a specified subclass.

-t *Type* Specifies a device type name. This flag can be used to restrict output to devices of a specified type.

Examples

1. To list all devices in the Predefined Devices object class with column headers, type:

   ```
   lsdev -P -H
   ```

   The system displays a message similar to the following:

   ```
   class          type           subclass   description
   logical_volume vgtype         vgsubclass Volume group
   logical_volume lvtype         lvsubclass Logical volume
   lvm            lvdd           lvm        LVM Device Driver
   posix_aio      posix_aio      node       Posix Asynchronous I/O
   aio            aio            node       Asynchronous I/O (Legacy)
   pty            pty            pty        Asynchronous Pseudo-Terminal
   mouse          030102         usbif      USB mouse
   keyboard       030101         usbif      USB keyboard
   driver         tmssar         node       Target Mode SSA Router
   tmssa          tmssa          tmssar     Target Mode SSA Device
   disk           hdisk          ssar       SSA Logical Disk Drive
   ```

```
pdisk           1000mbC         ssar        1GB SSA C Physical Disk Drive
pdisk           2000mbC         ssar        2GB SSA C Physical Disk Drive
.
.
.
disk            540mb2          scsi        540 MB SCSI Disk Drive
disk            540mb3          scsi        540 MB SCSI Disk Drive
disk            540mb4          scsi        540 MB SCSI Disk Drive
disk            540mb5          scsi        540 MB SCSI Disk Drive
disk            730mb2          scsi        730 MB SCSI Disk Drive
disk            810mb           scsi        810 MB SCSI Disk Drive
disk            810mb2          scsi        810 MB SCSI Disk Drive
bus             pcic            pci         PCI Bus
bus             isac            pci         ISA Bus
adapter         df1000f9        pci         FC Adapter
adapter         df1000f7        pci         FC Adapter
driver          efscsi          iocb        FC SCSI I/O Controller
Protocol Device
adapter         c1110358        pci         USB OHCI Adapter (c1110358)
adapter         ad100501        pci         ATA/IDE Controller Device
adapter         4f111100        pci         IBM 8-Port EIA-232/RS-422A (PCI) Adapter
adapter         ccm             pci         Name of the Common Character Mode device driver
driver          hdlc            331121b9    IBM HDLC Network Device
Driver
adapter         331121b9        pci         IBM 2-Port Multiprotocol Adapter (331121b9)
adapter         2b102005        pci         GXT130P Graphics Adapter
adapter         2b101a05        pci         GXT120P Graphics Adapter
adapter         23100020        pci         IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
.
.
.
if              tr              TR          Token Ring Network Interface
if              vi              VI          Virtual IP Address Network Interface
if              xt              XT          X.25 Network Interface
tcpip           inet            TCPIP       Internet Network Extension
swap            paging          nfs         NFS Swap DEVICE
drawer          media1          media       SCSI Device Drawer
drawer          scsi1           dasd        SCSI DASD Drawer
adapter         4f111b00        pci         IBM 128-Port Async (PCI)
Adapter
concentrator    16c232          sync_pci    16-Port RAN EIA-232 for 128-Port Adapter
concentrator    16e232          sync_pci    16-Port Enhanced RAN EIA-232 for 128-Port Adapter
concentrator    16e422          sync_pci    16-Port Enhanced RAN RS-422 for 128-Port Adapter
if              at              AT          ATM Network Interface
adapter         14105300        pci         IBM PCI 25MBPS ATM Adapter (14105300)
```

2. To list all the devices in the Customized Devices object class, type:

```
lsdev -C
```

The system displays a message similar to the following:

```
sys0       Available              System Object
sysplanar0 Available              System Planar
mem0       Available              Memory
L2cache0   Available              L2 Cache
proc0      Available 00-00        Processor
pci0       Available              PCI Bus
pci1       Available              PCI Bus
isa0       Available 10-58        ISA Bus
siota0     Available 01-Q1        Tablet Adapter
ppa0       Available 01-R1        CHRP IEEE1284 (ECP) Parallel Port Adapter
sa0        Available 01-S1        Standard I/O Serial Port
sa1        Available 01-S2        Standard I/O Serial Port
paud0      Available 01-Q2        Ultimedia Integrated Audio
siokma0    Available 01-K1        Keyboard/Mouse Adapter
fda0       Available 01-D1        Standard I/O Diskette Adapter
scsi0      Available 10-60        Wide/Ultra-2 SCSI I/O Controller
```

```
scsi1      Available 10-61        Wide/Ultra-2 SCSI I/O Controller
sa2        Available 10-68        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
sa3        Available 10-70        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
sa4        Available 10-78        IBM 8-Port EIA-232/RS-422A (PCI) Adapter
.
.
hd3        Defined                Logical volume
hd1        Defined                Logical volume
hd10opt    Defined                Logical volume
inet0      Available              Internet Network Extension
en0        Available 10-80        Standard Ethernet Network Interface
et0        Defined   10-80        IEEE 802.3 Ethernet Network Interface
lo0        Available              Loopback Network Interface
pty0       Available              Asynchronous Pseudo-Terminal
gxme0      Defined                Graphics Data Transfer Assist Subsystem
rcm0       Available              Rendering Context Manager Subsystem
aio0       Defined                Asynchronous I/O (Legacy)
posix_aio0 Defined                Posix Asynchronous I/O
tty0       Available 01-S1-00-00  Asynchronous Terminal
tty1       Available 01-S2-00-00  Asynchronous Terminal
```

3. To list the adapters that are in the Available state in the Customized Devices object class, type:

   ```
   lsdev -C -c adapter -S a
   ```

   The system displays a message similar to the following:

   ```
   sa0     Available 01-S1    Standard I/O Serial Port
   sa1     Available 01-S2    Standard I/O Serial Port
   siokma0 Available 01-K1    Keyboard/Mouse Adapter
   fda0    Available 01-D1    Standard I/O Diskette Adapter
   scsi0   Available 10-60    Wide/Fast-20 SCSI I/O Controller
   fcs0    Available 10-68    FC Adapter
   scsi1   Available 10-88    Wide/Ultra-2 SCSI I/O Controller
   fcs1    Available 20-60    FC Adapter
   sioka0  Available 01-K1-00 Keyboard Adapter
   siota0  Available 01-Q1    Tablet Adapter
   ppa0    Available 01-R1    CHRP IEEE1284 (ECP) Parallel Port Adapter
   paud0   Available 01-Q2    Ultimedia Integrated Audio
   tok0    Available 10-70    IBM PCI Tokenring Adapter (14101800)
   ent0    Available 10-80    IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
   sioma0  Available 01-K1-01 Mouse Adapter
   ```

4. To list all tape devices in the Predefined Devices object class, type:

   ```
   lsdev -P -c tape
   ```

   The system displays a message similar to the following:

   ```
   tape ost      fcp   Other FC SCSI Tape Drive
   tape scsd     fcp   FC SCSI Tape Drive
   tape ost      iscsi Other iSCSI Tape Drive
   tape scsd     iscsi iSCSI Tape Drive
   tape 1200mb-c scsi  1.2 GB 1/4-Inch Tape Drive
   tape 150mb    scsi  150 MB 1/4-Inch Tape Drive
   tape 3490e    scsi  3490E Autoloading Tape Drive
   tape 4mm2gb   scsi  2.0 GB 4mm Tape Drive
   tape 4mm4gb   scsi  4.0 GB 4mm Tape Drive
   tape 525mb    scsi  525 MB 1/4-Inch Tape Drive
   tape 8mm      scsi  2.3 GB 8mm Tape Drive
   tape 8mm5gb   scsi  5.0 GB 8mm Tape Drive
   tape 8mm7gb   scsi  7.0 GB 8mm Tape Drive
   tape 9trk     scsi  1/2-inch 9-Track Tape Drive
   tape ost      scsi  Other SCSI Tape Drive
   tape scsd     scsi  SCSI Tape Drive
   tape 4mm2gb2  scsi  2.0 GB 4mm Tape Drive
   ```

5. To list the supported device classes from the Predefined Devices object class, type:

   ```
   lsdev -P -r class
   ```

The system displays a message similar to the following:

```
PCM
adapter
aio
array
bus
cdrom
concentrator
container
dial
disk
diskette
drawer
driver
gxme
if
keyboard
lft
logical_volume
lpfk
lvm
memory
mouse
pdisk
planar
port
posix_aio
printer
processor
pseudo
pty
rcm
rwoptical
swap
sys
tablet
tape
tcpip
tmscsi
tmssa
tty
```

6. To list the supported subclasses in the Predefined Devices object class for the **disk** class, type:

   ```
   lsdev -P -c disk -r subclass
   ```

   The system displays a message similar to the following:

   ```
   dar
   fcp
   fdar
   ide
   iscsi
   scraid
   scsi
   ssar
   ```

7. To list the name, class, subclass, and type of every device in the Available state in the Customized Devices object class with column headers, type:

   ```
   lsdev -C -H -S a -F 'name class subclass type'
   ```

   The system displays a message similar to the following:

   ```
   name       class     subclass type
   sys0       sys       node     chrp
   sysplanar0 planar    sys      sysplanar_rspc
   mem0       memory    sys      totmem
   L2cache0   memory    sys      L2cache_rspc
   ```

```
proc0      processor sys      proc_rspc
pci0       bus       chrp     pci
pci1       bus       chrp     pci
isa0       bus       pci      isac
siota0     adapter   isa_sio  isa_tablet
ppa0       adapter   isa_sio  chrp_ecp
sa0        adapter   isa_sio  pnp501
sa1        adapter   isa_sio  pnp501
paud0      adapter   isa_sio  baud4232
siokma0    adapter   isa_sio  kma_chrp
fda0       adapter   isa_sio  pnp700
scsi0      adapter   pci      sym896
scsi1      adapter   pci      sym896
sa2        adapter   pci      4f111100
sa3        adapter   pci      4f111100
sa4        adapter   pci      4f111100
ent0       adapter   pci      23100020
mg20       adapter   pci      2b102005
sa5        adapter   pci      4f111100
sioka0     adapter   kma_chrp keyboard
sioma0     adapter   kma_chrp mouse
fd0        diskette  siofd    fd
cd0        cdrom     scsi     scsd
hdisk0     disk      scsi     scsd
kbd0       keyboard  std_k    ps2
mouse0     mouse     std_m    mse_3b
lvdd       lvm       lvm      lvdd
lft0       lft       node     lft
inet0      tcpip     TCPIP    inet
en0        if        EN       en
lo0        if        LO       lo
pty0       pty       pty      pty
rcm0       rcm       node     rcm
tty0       tty       rs232    tty
tty1       tty       rs232    tty
```

8. To list the name, class, location, and physloc of all adapter devices Customized Devices object class with column headers, type:

   ```
   lsdev -C -c adapter -F 'name class location physloc'
   ```

   The system displays a message similar to the following:

   ```
   siota0  adapter 01-Q1    P1/Q1
   ppa0    adapter 01-R1    P1/R1
   sa0     adapter 01-S1    P1/S1
   sa1     adapter 01-S2    P1/S2
   paud0   adapter 01-Q2    P1/Q2
   siokma0 adapter 01-K1    P1/K1
   fda0    adapter 01-D1    P1/D1
   scsi0   adapter 10-60    P1/Z1
   scsi1   adapter 10-61    P1/Z2
   sa2     adapter 10-68    P1-I3/Q1
   sa3     adapter 10-70    P1-I4/Q1
   sa4     adapter 10-78    P1-I5/Q1
   ent0    adapter 10-80    P1/E1
   mg20    adapter 10-88    P1-I6/G1
   sa5     adapter 20-58    P1-I1/Q1
   sioka0  adapter 01-K1-00 P1/K1
   sioma0  adapter 01-K1-01 P1/K1
   ```

9. To list the all the devices whose names begin with the letters 'hdi', enter:

   ```
   lsdev -l hdi*
   ```

   The system displays a message similar to the following:

   ```
   hdisk0 Available 10-60-00-8,0 16 Bit SCSI Disk Drive
   hdisk1 Available 10-60-00-9,0 16 Bit SCSI Disk Drive
   ```

10. To list the all the children of the `pci0` bus, type:

    ```
    lsdev -p pci0
    ```

    The system displays a message similar to the following:

    ```
    ent0  Available 10-80 IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
    isa0  Available 10-58 ISA Bus
    scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
    scsi1 Available 10-88 Wide/Ultra-2 SCSI I/O Controller
    tok0  Available 10-70 IBM PCI Tokenring Adapter (14103e00)
    ```

11. To list the devices whose names are contained in the file `/tmp/f`, type:

    ```
    cat /tmp/f | lsdev -l -
    ```

    The system displays a message similar to the following:

    ```
    pci0 Available  PCI Bus
    scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
    hdisk0 Available 10-60-00-8,0 16 Bit SCSI Disk Drive
    ```

Files

/usr/sbin/lsdev	Contains the **lsdev** command.

Related Information

The **chdev** command, **lsattr** command, **lsconn** command, **lsparent** command, **mkdev** command, **rmdev** command.

Devices Overview for System Management in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* provides information about adding, changing, moving, and removing devices.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* tells you about the SMIT application.

---

lsdisp Command

Purpose

Lists the displays available on the system.

Syntax

lsdisp [-l]

Description

The **lsdisp** command lists the displays currently available on the system, displaying a logical name of the display, a physical slot number of a display adapter, the type of bus to which a graphics display is attached, a display name and a description of each of the displays. This command also lists the default display.

Flags

-l	Specifies the removal of all header information and `Default display' from format.

Examples

To list all available displays, enter:

```
lsdisp
```

The following output of the **lsdisp** command lists three available displays for AIX 5.1 and earlier:

```
DEV_NAME  SLOT    BUS  ADPT_NAME  DESCRIPTION

ppr0      00-01   mca  POWER_G4   Midrange Graphics Adapter
gda0      00-03   mca  colordga   Color Graphics Display Adapter
ppr1      00-04   mca  POWER_Gt3  Midrange Entry Graphics Adapter

Default display = gda0
```

Files

bin/lsdisp Contains the **lsdisp** command.

Related Information

The **chdisp** command.

Low Function Terminal (LFT) Subsystem Overview in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

---

lsfilt Command

Purpose

Lists filter rules from either the filter table or the IP Security subsystem.

Syntax

lsfilt -v 4|6 [**-n** *fid_list*] [**-a**] [**-d**]

Description

Use the **lsfilt** command to list filter rules and their status.

Note: Filter description fields are not listed in the kernel. No filter description text will be displayed when active or dynamic filter rules are listed.

Flags

-a List only the active filter rules. The active filter rules are the rules being used by the filter kernel currently. If omitted, all the filter rules in the filter rule table will be listed.

-d Lists the dynamic filter rules used for Internet Key Exchange (IKE) tunnels. This table is built dynamically as IKE negotiations start creating IP Security tunnels and their corresponding filter rules are added to the dynamic IKE filter table.

-n Specifies the ID(s) of filter rule(s) that are displayed. The *fid_list* is a list of filter IDs separated by a space or ″**,**″ or ″**-**″. The **-n** is not for active filter rules. This flag cannot be used with the **-a** flag.

-v IP version of the filter rule you want to list. Valid values for this flag are **4** and **6**. If this flag is not used, both IP version 4 and IP version 6 are listed.

lsfont Command

Purpose

Lists the fonts available to the display.

Syntax

lsfont [**-l**]

Description

The **lsfont** command displays a list of the fonts available to the display. The font identifier can help you change fonts using the **chfont** command.

You can use a 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 lsfont** fast path to run this command.

Flags

-l Specifies the removal of all header information from format of data.

Examples

To list all fonts available to the display, enter:

```
lsfont
```

The following example displays the font identifier, font name, glyph size, and font encoding for each available font:

```
FONT  FILE            GLYPH  FONT
ID    NAME            SIZE   ENCODING
====  ==============  =====  =========
0     Erg22.iso1.snf  12x30  ISO8859-1
1     Erg11.iso1.snf   8x15  ISO8859-1
```

For further details about the fonts available, see Text Fonts for the AIX System.

Files

/bin/lsfont	Contains the **lsfont** command.
/usr/lpp/fonts	Contains fonts directory.

Related Information

The **chfont** command, **mkfont** command.

Low Function Terminal (LFT) Subsystem Overview in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

lsfs Command

Purpose

Displays the characteristics of file systems.

Syntax

lsfs [**-q**] [**-c** | **-l**] [**-a** | **-v** *VfsType* | **-u** *MountGroup*| [*FileSystem*...]]

Description

The **lsfs** command displays characteristics of file systems, such as mount points, automatic mounts, permissions, and file system size. The *FileSystem* parameter reports on a specific file system. The following subsets can be queried for a listing of characteristics:

- All file systems
- All file systems of a certain mount group
- All file systems of a certain virtual file system type
- One or more individual file systems

The **lsfs** command displays additional Journaled File System (JFS) or Enhanced Journaled File System (JFS2) characteristics if the **-q** flag is specified.

You can use a Web-based System Manager File Systems application (**wsm fs** fast path) to run this command. You could also use the System Management Interface Tool (SMIT) **smit lsfs** fast path to run this command.

Flags

-a	Lists all file systems (default).
-c	Specifies that the output should be in colon format.
-l	Specifies that the output should be in list format.
-q	Queries the logical volume manager (LVM) for the logical volume size (in 512-byte blocks). Additional information is provided for JFS and JFS2 file systems. Queries the JFS superblock for the file system size, the fragment size, the compression algorithm (if any), and the number of bytes per i-node (nbpi). Queries the JFS2 superblock for the file system size, the file system block size, and sparse file support. This information is not reported for other virtual file system types. It is displayed in addition to other file system characteristics reported by the **lsfs** command.
-u *MountGroup*	Reports on all file systems of a specified mount group.
-v *VfsType*	Reports on all file systems of a specified type.

Examples

1. To show all file systems in the **/etc/filesystems** file, enter:

   ```
   lsfs
   ```

2. To show all file systems of vfs type `jfs`, enter:

   ```
   lsfs -v jfs
   ```

3. To show the file system size, the fragment size, the compression algorithm (if any), and the number of bytes per i-node as recorded in the superblock of the root file system, enter:

```
lsfs -q /
```

Files

/etc/filesystems	Lists the known file systems and defines their characteristics.

Related Information

The **chfs** command, **crfs** command, **rmfs** command.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

Mounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains mounting files and directories, mount points, and automatic mounts.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains SMIT structure, main menus, and tasks.

---

lsgroup Command

Purpose

Displays group attributes.

Syntax

lsgroup [**-R** *load_module*] [**-c** | **-f**] [**-a** *List*] {**ALL** | *Group* [**,***Group*] ...}

Description

The **lsgroup** command displays group attributes. You can use this command to list all the system groups and their attributes or you can list all the attributes of individual groups. Since there is no default parameter, you must enter the **ALL** keyword to list all the system groups and their attributes. All the attributes described in the **chgroup** command appear. If the **lsgroup** command cannot read one or more attributes, it lists as much information as possible. To view a selected attribute, use the **-a** *List* flag.

> **Note:** If you have a Network Information Service (NIS) database installed on your system, some user information may not appear when you use the **lsgroup** command.

By default, the **lsgroup** command lists each group on one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the group attributes in stanza format, use the **-f** flag. To list the information in colon-separated records, use the **-c** flag.

You can use a Web-based System Manager Users application (**wsm users** fast path) to run this command. You could also use the System Management Interface Tool (SMIT) **smit lsgroup** fast path to run this command.

Flags

-a *List*	Specifies the attributes to display. The *List* parameter can include any attribute defined in the **chgroup** command, and requires a blank space between attributes. If you specify an empty list, only the group names are listed.

-c Displays the attributes for each group in colon-separated records, as follows:

```
#name:  attribute1:  attribute2:  ...
Group:  value1:      value2:      ...
```

-f Displays the group attributes in stanzas. Each stanza is identified by a group name. Each *Attribute*=*Value* pair is listed on a separate line:

```
group:
      attribute1=value
      attribute2=value
      attribute3=value
```

-R *load_module* Specifies the loadable I&A module used to get the group attribute list.

Security

Access Control: This command should be a general user program with execute (x) access for all users. Attributes are read with the access rights of the invoker, so all users may not be able to access all the information. This depends on the access policy of your system. This command should have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/group**
r	**/etc/security/group**
r	**/etc/passwd**

Limitations

Listing a group may not be supported by all loadable I&A modules. If the loadable I&A module does not support listing a group, then an error is returned.

Examples

1. To display the attributes of the `finance` group in the default format, type:

   ```
   lsgroup finance
   ```

2. To display the `id`, members (`users`), and administrators (`adms`) of the `finance` group in stanza format, type:

   ```
   lsgroup  -f  -a id users adms finance
   ```

3. To display the attributes of all the groups in colon-separated format, type:

   ```
   lsgroup  -c ALL
   ```

   All the attribute information appears, with each attribute separated by a blank space.

4. To display the attributes of the LDAP I&A loadable module group monsters, type:

   ```
   lsgroup -R LDAP monsters
   ```

Files

/usr/sbin/lsgroup	Contains the **lsgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/passwd	Contains user IDs, user names, home directories, login shell, and finger information.

Related Information

The **chfn** command, **chgroup** command, **chgrpmem** command, **chsh** command, **chuser** command, **lsuser** command, **mkgroup** command, **mkuser** command, **passwd** command, **pwdadm** command, **rmgroup** command, **rmuser** command, **setgroups** command, **setsenv** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

lsitab Command

Purpose

Lists records in the **/etc/inittab** file.

Syntax

lsitab { **-a** | *Identifier* }

Description

The **lsitab** command displays a record in the **/etc/inittab** file. You can display all of the records in the **/etc/inittab** file, or use the *Identifier* parameter to display a specific record. The *Identifier* parameter is a 14-character field that uniquely identifies an object.

Flags

-a Specifies that all records in the **/etc/inittab** file are listed.

Examples

1. To display the record for `tty2`, enter:
   ```
   lsitab "tty002"
   ```
   The output is similar to: `tty002:2:respawn:/usr/sbin/getty /dev/tty2`
2. To display all of the records in the **/etc/inittab** file, enter:
   ```
   lsitab -a
   ```
   All of the records in the **/etc/inittab** file are displayed.

Files

/etc/inittab Indicates which processes the **init** command starts.

Related Information

The **chitab** command, **init** command, **mkitab** command, **rmitab** command.

---

lskbd Command

Purpose

List the current software keyboard map loaded into the system.

Syntax

lskbd

Description

The **lskbd** command displays the absolute pathname of the current software keyboard map loaded into the system.

To list the current software keyboard map enter:

```
lskbd
```

You can use the Devices application in Web-based System Manager (wsm) to change device characteristics. You could also use the System Management Interface Tool (SMIT) **smit lskbd** fast path to run this command.

Note: This command can be used only on an LFT display.

Example

Following is an example of the listing displayed by the lskbd command:

```
The current software keyboard map = /usr/lib/nls/loc/C.lftkeymap
```

Files

/usr/bin/lskbd	Contains the **lskbd** command.
/usr/lib/nls/loc	Software keyboard map directory.

Related Information

The **chkbd** command, **smit** command.

Low Function Terminal (LFT) Subsystem Overview in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

lslicense Command

Purpose

Displays the number of fixed licenses and the status of the floating licensing.

Syntax

lslicense [**-A**] [**-c**]

Description

The **lslicense** command displays the number of fixed licenses and the status of the floating licensing.

Flags

-A The **-A** flag causes the **lslicense** command to report the current number of available fixed licenses. When the **-A** flag is not specified, the maximum number of fixed licenses and license status is reported.

-c Displays the output in : (colon) form.

Examples

1. To display the number of fixed licenses and the floating license status, enter:

   ```
   lslicense
   ```

   Example output would be:

   ```
   Maximum number of fixed licenses is 10.
   Floating licensing is enabled.
   ```

2. To display the number of fixed licenses and the floating license status in a colon format, enter:

   ```
   lslicense  -c
   ```

   Example output would be:

   ```
   #fixed:floating
   10:on
   ```

3. To display license information including the number of available fixed licenses, enter:

   ```
   lslicense -A
   ```

   Ouput similar to the following will display:

   ```
   Maximum number of fixed licenses is 2.
   Floating licensing is disabled.
   Number of available fixed licenses is 2.
   ```

Related Information

The **chlicense** command and **monitord** daemon.

---

lslpp Command

Purpose

Lists installed software products.

Syntax

lslpp { **-d** | **-E** | **-f** | **-h** | **-i** | **-l** | **-L** | **-p** }] [**-a**] [**-c**] [**-J**] [**-q**] [**-l**] [**-O** { [**r**] [**s**] [**u**] }] [[*FilesetName ...* | *FixID ...* | **all**]

lslpp -w [**-c**] [**-q**] [**-O** { [**r**] [**s**] [**u**] }] [*FileName ...* | **all**]

lslpp -L -c [**-v**]

lslpp -S [**A**|**O**]

lslpp -e

Description

The **lslpp** command displays information about installed filesets or fileset updates. The *FilesetName* parameter is the name of a software product. The *FixID* (also known as PTF or program temporary fix ID) parameter specifies the identifier of an update to a formatted fileset.

When only the **-l** (lowercase L) flag is entered, the **lslpp** command displays the latest installed level of the fileset specified for formatted filesets. The base level fileset is displayed for formatted filesets. When the **-a** flag is entered along with the **-l** flag, the **lslpp** command displays information about all installed filesets for the *FilesetName* specified. The **-l** (uppercase i) flag combined with the **-l** (lowercase L) flag specifies that the output from the **lslpp** command should be limited to base level filesets.

The **-d**, **-f**, **-h**, **-i**, **-l** (lowercase L), **-L**, and **-p** flags request different types of output reports.

The **-a**, **-c**, **-J**, and **-q** flags specify the amount and format of the information that is displayed in the report.

The **-O** flag specifies that data is to come from a specified part of the fileset. The part may be the root part, **-Or**, the share part, **-Os**, or the **usr** part, **-Ou**.

The default value for the *FilesetName* parameter is **all**, which displays information about all installed software products. Pattern matching characters, such as * (asterisk) and ? (question mark), are valid in the *ProductName* and *FixID* parameters. You don't have to enclose these characters in '' (single quotation marks). However, using single quotation marks prevents you from searching the contents of your present directory.

Output Values

Much of the output from the **lslpp** command is understandable without an explanation. Other fields contain data that needs to be defined. The following sections define terms used in several of the output fields.

State Values

The **state** field in the **lslpp** output gives the state of the fileset on your system. It can have the following values:

State	Definition
APPLIED	The specified fileset is installed on the system. The **APPLIED** state means that the fileset can be rejected with the **installp** command and the previous level of the fileset restored. This state is only valid for Version 4 fileset updates and 3.2 migrated filesets.
APPLYING	An attempt was made to apply the specified fileset, but it did not complete successfully, and cleanup was not performed.
BROKEN	The specified fileset or fileset update is broken and should be reinstalled before being used.
COMMITTED	The specified fileset is installed on the system. The **COMMITTED** state means that a commitment has been made to this level of the software. A committed fileset update cannot be rejected, but a committed fileset base level and its updates (regardless of state) can be removed or deinstalled by the **installp** command.
EFIX LOCKED	The specified fileset was installed sucessfully and locked by the emergency fix (efix) manager.
OBSOLETE	The specified fileset was installed with an earlier version of the operating system but has been replaced by a repackaged (renamed) newer version. Some of the files that belonged to this fileset have been replaced by versions from the repackaged fileset.
COMMITTING	An attempt was made to commit the specified fileset, but it did not complete successfully, and cleanup was not performed.
REJECTING	An attempt was made to reject the specified fileset, but it did not complete successfully, and cleanup was not performed.

Action Values

The **action** field in the **lslpp** output identifies the installation action that was taken for the fileset. The following values may be found in this field:

Action	Definition
APPLY	An attempt was made to apply the specified fileset.
CLEANUP	An attempt was made to perform cleanup for the specified fileset.
COMMIT	An attempt was made to commit the specified fileset.
REJECT	An attempt was made to reject the specified fileset.

Status Values

The **status** field in the **lslpp** output identifies the resultant status in the history of installation actions. The following values may be found in this field:

Status	Definition
BROKEN	The fileset was left in a broken state after the specified action.
CANCELED	The specified action was canceled before it completed.
COMPLETE	The commitment of the fileset has completed successfully.

Flags

-a Displays **all** the information about filesets specified when combined with other flags. This flag shows all updates when combined with the **-l** flag and all history when combined with the **-h** flag. This flag cannot be specified with the **-f** flag.

-c Displays information as a list separated by colons. This flag cannot be specified with the **-J** flag.

-d Displays filesets that are dependents of the specified software. A dependent fileset is one that has the specified software as a prerequisite, corequisite, ifrequisite, or installed requisite.

-e Displays every efix installed on the system.

-E Lists license agreements.

-f Displays the names of the files added to the system during installation of the specified fileset. This flag cannot be specified with the **-a** flag.

-h Displays the installation and update history information for the specified fileset. You cannot use this flag with the **-J** flag.

-I (uppercase i) Limits the inputs to software products.

-i Displays the product information for the specified fileset.

-J Generates output in a form suitable for the System Management Interface Tool (SMIT) command to list output. This flag can only be specified with the **-l** (lowercase L) and **-L** flags.

-l (lowercase L) Displays the name, most recent level, state, and description of the specified fileset.

-L Displays the name, most recent level, state, and description of the specified fileset. Part information (**usr**, **root**, and **share**) is consolidated into the same listing. For formatted filesets, it displays the most recent maintenance level for the specified filesets. In addition, this flag lists any subsystem selective fixes that were installed on top of the maintenance level. RPM and ISMP images are also listed.

-O Lists information for the specified part of the fileset. When the **-O** flag is not specified information is listed for all parts. This option is designed for use by the **nim** command to list software product information for diskless or dataless workstations. You can use the following flags with this flag:

- **-r** Indicates to list information for the root part.
- **-s** Indicates to list information for the **/usr/share** part.
- **-u** Indicates to list information for the **/usr** part.

-p Displays requisite information for the specified fileset.

-q Suppresses the display of column headings.

-S [**A**|**O**]	Displays a list of automatically installed filesets and a list of optionally installed filesets. If the **-S** flag is followed by **A**, then only the automatically installed filesets are listed. If the **-S** flag is followed by **O**, then only the optionally installed filesets are listed.
-v	Displays additional information from the vendor database. This flag is only valid when used with both the **-L** and the **-c** flag.
-w	Lists fileset that owns this file. This flag applies to AIX 4.2 or later.

You must specify one of the mutually exclusive flags: **-d**, **-e**, **-E**, **-f**, **-h**, **-i**, **-l**, **-L**, **-p**, **-s**, and **-w**.

Examples

1. To list the installation state for the most recent level of installed filesets for all of the **bos.rte** filesets, type:

   ```
   lslpp -l "bos.rte.*"
   ```

2. To list the installation state for the base level and updates for the fileset **bos.rte.filesystem**, type:

   ```
   lslpp -La bos.rte.filesystem
   ```

3. To list the installation history information of all the filesets in the **bos.net** software package, type:

   ```
   lslpp -ha 'bos.net.*'
   ```

4. To list the names of all the files of the **bos.rte.lvm** fileset, type:

   ```
   lslpp -f bos.rte.lvm
   ```

5. To list the fileset that owns **installp**, type:

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

   Output similar to the following displays:

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

6. To list the fileset that owns all file names that contain `installp`, type:

   ```
   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
   ```

7. To display all files in the inventory database, type:

   ```
   lslpp -w
   ```

8. To display the installation state for the RPM `cdrecord` image, type:

   ```
   lslpp -L cdrecord
   ```

Files

/etc/objrepos/history	Specifies installation and update history information of all software products on the root.
/usr/lib/objrepos/history	Specifies installation and update history information of all software products on the **/usr** file system.

/usr/share/lib/objrepos/history	Specifies installation and update history information of all software products on the **/usr/share** file system.
/etc/objrepos/lpp	Specifies installation information of all software products on the root.
/usr/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr** file system.
/usr/share/lib/objrepos/lpp	Specifies installation information of all software products on the **/usr/share** file system.
/etc/objrepos/product	Specifies installation and update information of all software products on the root.
/usr/lib/objrepos/product	Specifies installation and update information of all software products on the **/usr** file system.
/usr/share/lib/objrepos/product	Specifies installation and update information of all the software products on the **/usr/share** file system.
/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 **emgr** command, **installp** command, **inulag** command, **nim** command.

---

lslv Command

Purpose

Displays information about a logical volume.

Syntax

To Display Logical Volume Information

lslv [**-L**] [**-l**| **-m**] [**-n***PhysicalVolume*] *LogicalVolume*

To Display Logical Volume Allocation Map

lslv [**-L**] [**-n***PhysicalVolume*] **-p***PhysicalVolume* [*LogicalVolume*]

Description

The **lslv** command displays the characteristics and status of the *LogicalVolume* or lists the logical volume allocation map for the physical partitions on the *PhysicalVolume*. The logical volume can be a name or identifier.

Note: If the **lslv** command cannot find information for a field in the Device Configuration Database, it will insert a question mark (?) in the value field. As an example, if there is no information for the LABEL field, the following is displayed:

```
LABEL: ?
```

The command attempts to obtain as much information as possible from the description area when it is given a logical volume identifier.

You can use the Volumes application in Web-based System Manager to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit lslv** fast path to run this command.

Flags

-L Specifies no waiting to obtain a lock on the Volume group.
Note: If the volume group is being changed, using the **-L** flag gives unreliable date.

-l Lists the following fields for each physical volume in the logical volume:

PV Physical volume name.

Copies The following three fields:

- The number of logical partitions containing at least one physical partition (no copies) on the physical volume
- The number of logical partitions containing at least two physical partitions (one copy) on the physical volume
- The number of logical partitions containing three physical partitions (two copies) on the physical volume

In band
The percentage of physical partitions on the physical volume that belong to the logical volume and were allocated within the physical volume region specified by Intra-physical allocation policy.

Distribution
The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

-m Lists the following fields for each logical partition:

LPs Logical partition number.

PV1 Physical volume name where the logical partition's first physical partition is located.

PP1 First physical partition number allocated to the logical partition.

PV2 Physical volume name where the logical partition's second physical partition (first copy) is located.

PP2 Second physical partition number allocated to the logical partition.

PV3 Physical volume name where the logical partition's third physical partition (second copy) is located.

PP3 Third physical partition number allocated to the logical partition.

-n *PhysicalVolume* Accesses information from the specific descriptor area of *PhysicalVolume* variable. The information may not be current since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the validated information is accessed and therefore the information that is displayed is current. The volume group need not be active when you use this flag.

-p *PhysicalVolume* Displays the logical volume allocation map for the *PhysicalVolume* variable. If you use the *LogicalVolume* parameter, any partition allocated to that logical volume is listed by logical partition number. Otherwise, the state of the partition is listed as one of the following:

used Indicates that the partition is allocated to another logical volume.

free Indicates that the specified partition is not being used on the system.

stale Indicates that the specified partition is no longer consistent with other partitions. The computer lists the logical partitions number with a question mark if the partition is stale.

If no flags are specified, the following status is displayed:

`Logical volume`	Name of the logical volume. Logical volume names must be unique systemwide and can range from 1 to 15 characters.
`Volume group`	Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.
`Logical volume identifier`	Identifier of the logical volume.
`Permission`	Access permission; `read-only` or `read-write`.
`Volume group state`	State of the volume group. If the volume group is activated with the **varyonvg** command, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating all physical volumes are not active). If the volume group is not activated with the **varyonvg** command, the state is `inactive`.
`Logical volume state`	State of the logical volume. The `Opened/stale` status indicates the logical volume is open but contains physical partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.
`Type`	Logical volume type.
`Write verify`	Write verify state of On or Off.
`Mirror write consistency`	Mirror write consistency state of Yes or No.
`Max LPs`	Maximum number of logical partitions the logical volume can hold.
`PP size`	Size of each physical partition.
`Copies`	Number of physical partitions created for each logical partition when allocating.
`Schedule policy`	Sequential or parallel scheduling policy.
`LPs`	Number of logical partitions currently in the logical volume.
`PPs`	Number of physical partitions currently in the logical volume.
`Stale partitions`	Number of physical partitions in the logical volume that are not current.
`Bad blocks`	Bad block relocation policy.
`Inter-policy`	Inter-physical allocation policy.
`Strictness`	Current state of allocation, strict, nonstrict, or superstrict. A strict allocation states that no copies for a logical partition are allocated on the same physical volume. If the allocation does not follow the strict criteria, is is called nonstrict. A nonstrict allocation states that at least one occurrence of two physical partitions belong to the same logical partition. A superstrict allocation states that no partition from one mirror copy may reside the same disk as another mirror copy.
`Intra-policy`	Intra-physical allocation policy.
`Upper bound`	If the logical volume is super strict, upper bound is the maximum number of disks in a mirror copy.
`Relocatable`	Indicates whether the partitions can be relocated if a reorganization of partition allocation takes place.
`Mount point`	File system mount point for the logical volume, if applicable.
`Label`	Specifies the label field for the logical volume.
`PV distribution`	The distribution of the logical volume within the volume group. The physical volumes used, the number of logical partitions on each physical volume, and the number of physical partitions on each physical volume are shown.
`striping width`	The number of physical volumes being striped across.
`strip size`	The number of bytes per stripe.

Examples

1. To display information about logical volume `lv03`, enter:

   ```
   lslv lv03
   ```

Information about logical volume `lv03`, its logical and physical partitions, and the volume group to which it belongs is displayed.

2. To display the logical volume allocation map for `hdisk2`, enter:

   ```
   lslv -p hdisk2
   ```

   An allocation map for`hdisk2` is displayed, showing the state of each partition. Since no *LogicalVolume* parameter was included, the map does not contain logical partition numbers specific to any logical volume.

3. To display information about logical volume`lv03` by physical volume, enter:

   ```
   lslv -l lv03
   ```

   The characteristics and status of `lv03` are displayed, with the output arranged by physical volume.

4. To display information about physical volume `hdisk3` gathered from the descriptor area on `hdisk2`, enter:

   ```
   lslv -n hdisk2 -p hdisk3 lv02
   ```

   An allocation map, using the descriptor area on `hdisk2`, is displayed. Because the *LogicalVolume* parameter is included, the number of each logical partition allocated to that logical volume is displayed on the map.

5. To display information about a specific logical volume, using the identifier, enter:

   ```
   lslv 00000256a81634bc.2
   ```

   All available characteristics and status of this logical volume are displayed.

File

/usr/sbin Contains the **lslv** command.

Related Information

The **chlv** command**, lspv** command**, lsvg** command, **mklv** command, **reorgvg** command, **varyonvg** command.

Monitoring and Tuning Disk I/O in *AIX 5L Version 5.2 Performance Management Guide*

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the Logical Volume Manager, physical volumes, logical volumes, volume groups, organization, ensuring data integrity, and allocation characteristics.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the structure, main menus, and tasks that are done with SMIT.

lsmaster Command

Purpose

Displays the characteristics for the configuration of an NIS master server.

Syntax

/usr/sbin/lsmaster [**-c** | **-l**]

Description

The **lsmaster** command displays the characteristics of an NIS master server. The host names of the slave servers are listed along with the currently served domains.

You can use the Devices application in Web-based System Manager (wsm) to change device characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsmaster** fast path to run this command.

Flags

-c Specifies that the output should be in colon format. This flag is the default.
-l Specifies that the output should be in list format.

Examples

To list the NIS master server characteristics in colon format, enter:

```
lsmaster -c
```

Files

/var/yp/*domainname* directory	Contains the NIS maps for the NIS domain.

Related Information

The **chmaster** command, **mkmaster** command, **rmyp** command, **smit** command.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

lsmcode Command

Purpose

Displays microcode and firmware levels of the system and adapters and devices.

Syntax

lsmcode [**-A** | **-d** *Name*] [**-r** | **-c**]

Description

The **lsmcode** command when run without any flags, displays the platform system firmware microcode level and the service processor microcode levels, if supported. Not all systems contain a service processor, nor do all systems support displaying the system processor level. Information on a specific device is displayed with the **-d** flag.

If you run the **lsmcode** command with the **-r** or **-c** flag, it displays the microcode levels in a **printf** format; that is, not a menu. This method is preferred if running **lsmcode** from a script.

Flags

-A Displays microcode level information for all supported devices. Using this flag assumes the **-r** flag.

-c Displays the microcode/firmware levels without using menus.

-d *Name* Displays microcode level information for the named device.

-r Displays the microcode/firmware levels in a tabular format. The microcode level is preceded by a Type if supported or required.

Current supported Types are as follows:

system
System Firmware

service
Service Processor

adapter
Adapter Functional microcode

adapter-boot
Adapter Boot Microcode

raid-dasd
DASD Microcode in a RAID array

backplane
Backplane Microcode in a RAID subsystem

Examples

1. To display the system firmware level and service processor (if present), type:

   ```
   lsmcode -c
   ```

   The system displays a message similar to the following:

   ```
   System Firmware level is TCP99256
   ```

2. To display the system firmware level and service processor (if present) in raw mode, type:

   ```
   lsmcode -r
   ```

   The system displays a message similar to the following:

   ```
   system:TCP99256
   ```

3. To display the adapter microcode levels for a RAID adapter **scraid0**, type:

   ```
   lsmcode -r -d scraid0
   ```

   The system displays a message similar to the following:

   ```
   adapter:4.20.18|adapter-boot:4.00.26
   raid-dasd:22:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd:26:FFC #:DDYS-T0.524D3031.53393446
   raid-dasd:2e:FFC #:DDYS-T0.525A3034.53393243
   ```

4. To display the microcode level for a tape drive **rmt0**, type:

```
lsmcode -r -d rmt0
```

   The system displays a message similar to the following:

```
C009
```

5. To display the microcode level for all supported devices, type:

```
lsmcode -A
```

   The system displays a message similar to the following:

```
sys0!system:TCP99256
rmt0!C009
scraid0!adapter:4.20.18|adapter-boot:4.00.26
raid-dasd:22:FFC #:DDYS-T0.524D3031.53393446
raid-dasd:26:FFC #:DDYS-T0.524D3031.53393446
raid-dasd:2e:FFC #:DDYS-T0.525A3034.53393243
.....
```

Files

/usr/sbin/lsmcode	Contains the **lsmcode** command.

Related Information

The **diag** command.

---

lsmksysb Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

lsmksysb [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **lsmksysb** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **lsmksysb** command also works for multi-volume backups such as multiple CDs, DVDs, or tapes.

The **lsmksysb -r** and **restorevgfiles** commands perform identical operations and should be considered interchangeable.

Flags

-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You may need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.
-b *blocks*	Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read will default to 100.

-B Prints the volume group backup log to **stdout**.

This flag will display the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command used to make backup, date, shrink size, full size of the backup, and recommended maintenance level (if any).
Note: The shrink size is the size of the data on all filesystems. The full size is total size of each filesystem (unused + data).

-c Produces colon-separated output. This flag only works with the **-l** and **-L** flags.

-d *path* Specifies the directory path to which the files will be restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.

-D Produces debug output.

-l Displays useful information about a volume group backup.

This flag requires the **-f** *device* flag. This flag causes **lsmksysb** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all filesystems. The full size is the total size of each filesystem (unused + data). The **-l** flag also displays the logical volume and filesystem information of the backed up volume group, equivalent to running ″**lsvg -l** *vgname*″.

-L Displays lpp fileset information about a **mksysb** backup only.

This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking ″**lslpp -l**″ on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.

-f *device* Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.

-r Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup will be restored. If the **-r** flag is not used, then executing the **lsmksysb** command only lists the files in the specified backup.

-s Specifies that the backup source is a user volume group and not rootvg.

-V Verifies a tape backup.

This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **lsmksysb** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

file_list Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory should be specified in the space-separated list. All files in the specified directory will be restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup located on the default device **/dev/rmt0**, enter:
   ```
   lsmksysb
   ```
2. To list the contents of the system backup located on device **/dev/cd1**, enter:
   ```
   lsmksysb -f /dev/cd1
   ```
3. To list the contents of the system backup located on device **/dev/cd1**, which is a user volume group that is not rootvg, enter:
   ```
   lsmksysb -f /dev/cd1 -s
   ```

4. To restore **/etc/filesystems** from the system backup located on device **/dev/cd1**, enter:
   ```
   lsmksysb -f /dev/cd1 -r ./etc/filesystems
   ```
5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is located on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter:
   ```
   lsmksysb -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```
6. To display colon separated lpp information about a **mksysb** backup tape located on **/dev/rmt0**, enter the following:
   ```
   lsmksysb -Lc -f /dev/rmt0
   ```
7. To display the volume group backup log to **stdout**, enter:
   ```
   lssavevg -B
   ```
8. To list volume group and general backup data about a backup located at **/tmp/mybackup**, enter:
   ```
   lsmksysb -l -f /tmp/mybackup
   ```
9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter:
   ```
   lsmksysb -V -f /dev/rmt0
   ```

Files

/usr/bin/lsmksysb	Contains the **lsmksysb** command

Related Information

The **restorevgfiles** command.

---

lsnamsv Command

Purpose

Shows name service information stored in the database.

Syntax

lsnamsv { **-C** | **-S***"AttributeList ..."* } [**-Z**]

Description

The **lsnamsv** high-level command shows customized, TCP/IP-based name service information from the **/etc/resolv.conf** file only. No information from the name server database is shown. The command can extract all customized name service information or selected name service attribute information from the configuration database.

You can use the Network application in Web-based System Manager (wsm) to change network characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsnamerslv** fast path to run this command.

Flags

-C
Extracts all customized name service configuration information.

-S *"AttributeList ..."*
Specifies a selected set of attributes to be extracted from the system configuration database. Attributes can be the following:

domain Domain name

nameserver
Internet address of name server in dotted decimal format

-Z Specifies that the output be in colon format. This flag is used when the **lsnamsv** command is invoked from the SMIT usability interface.

Examples

1. To list all customized name service configuration information in dotted decimal format, enter the following command:

   ```
   lsnamsv -C
   ```

2. To list selected attributes, enter the following command:

   ```
   lsnamsv -S "domain nameserver"
   ```

   The **-S** flag indicates that the quoted list that follows contains a list of attributes to display.

Related Information

The **namerslv** command.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Naming and Understanding the SMIT Interface for TCP/IP in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

lsnfsexp Command

Purpose

Displays the characteristics of directories that are exported with the Network File System (NFS).

Syntax

/usr/sbin/lsnfsexp [**-c** | **-l**] [*Directory*] [**-f** *Exports_file*]

Description

The **lsnfsexp** command displays the characteristics of NFS-exported directories. The *Directory* parameter specifies the directory to be displayed. If no directory is specified, all directories exported with NFS will be displayed.

Flags

-c	Specifies that the output should be in colon format.
-l	(Lowercase L) Specifies that the output should be in list format. This flag is the default.
Directory	Specifies the directory to be displayed. If no directory is specified, all directories exported with NFS will be displayed.
-f *Exports_file*	Specifies the full path name of the export file to use if other than **/etc/exports**.

Examples

1. To list all of the directories currently exported with NFS in the colon format, enter:

   ```
   lsnfsexp -c
   ```

2. To list all of the directories currently exported with NFS in the colon format and use a specified path name other than **/etc/exports** enter:

   ```
   lsnfsexp -c -f /etc/exports.other
   ```

File

/etc/exports Lists the directories the server can export.

Related Information

The **chnfsexp** command, **exportfs** command, **mknfsexp** command, **rmnfsexp** command, **smit** command.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

---

lsnfsmnt Command

Purpose

Displays the characteristics of NFS mountable file systems.

Syntax

/usr/sbin/lsnfsmnt [**-c** | **-l**] [*FileSystem*]

Description

The **lsnfsmnt** command displays the current characteristics of NFS mountable file systems. The *FileSystem* parameter specifies the file system to be displayed in the output. If no file system is specified, all of the file systems that are NFS mountable will be displayed.

Flags

-c Specifies that the output should be in colon format.
-l (Lowercase L) Specifies that the output should be in list format. This flag is the default.

Examples

To list all of the NFS mounted file systems in the colon format, enter:

```
lsnfsmnt -c
```

Files

/etc/filesystems Centralizes file system characteristics.

Related Information

The **chnfsmnt** command, **mknfsmnt** command, **mount** command, **rmnfsmnt** command, **smit** command.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

---

lsnim Command

Purpose

Displays information about the Network Installation Management (NIM) environment.

Syntax

To Display a List of Supported NIM Classes, Subclasses, or Types

lsnim { **-p**| **-P**} [**-c***Class*| **-S**]

To Display Predefined NIM Information

lsnim { **-p**| **-P**} [**-c***Class*| **-s***Subclass*| **-t***Type*] [**-l**| **-O**] [**-Z**]

OR

lsnim { **-p**| **-P**}[**-a** *Attribute*] . . . [**-Z**]

To Display Attributes Required for an Operation

lsnim -t*Type* **-q***Operation*

To Display Information about All Customized NIM Objects

lsnim [**-c***Class*| **-s***Subclass*| **-t***Type*] [**-l**| **-O**] [**-Z**]

OR

lsnim [**-a***Attribute*] . . . [**-Z**]

To Display Information about a Specific NIM Object

lsnim[**-l** | **-O**| **-a** *Attribute* . . .][**-Z**] *ObjectName*

OR

lsnim[**-q***Operation*] *ObjectName*

To Display Information about Resources Available to a Specific NIM Machine

lsnim -L [**-s***Subclass*| **-t***Type*]*ObjectName*

To Display Information about NIM Groups

lsnim -g | **-m** [**-a** *Attribute* | **-c***Class* | **-L** | **-l** | **-s***Subclass* | **-t***Type*] *GroupObjectName*

Description

The **lsnim** command displays information about the NIM environment. This information is divided into two basic categories: predefined and customized.

Predefined information consists of values that are preset by NIM and cannot be modified by the user. Examples of predefined information include:

- The types of objects supported by NIM
- The classes and subclasses into which NIM organizes objects
- The operations that can be performed on NIM objects
- The attributes that can be entered by the user

In general, NIM uses this information to make decisions during operations. Predefined information can be displayed by using the **-p** or **-P** flag. The **-p** flag displays default values while the **-P** flag displays help information.

Customized information consists of values that you enter or modify. This information represents the physical environment in which NIM operates. Related pieces of customized information are grouped together to form *objects*, which are organized in the NIM database by object type and class. Some examples of object types include `diskless`, `paging`, and `standalone`. Two examples of object classes are `machines` and `network`.

For example, a standalone workstation that is part of the NIM environment is represented by a unique object. This object is classified by NIM as a `standalonemachines` object, where `standalone` represents the object type and `machines` represents the object class. Entering the **lsnim** command on the command line without any flags displays information on all customized objects.

You can also use the **lsnim** command to display relationships between customized objects. Choose an object to *anchor* on (specified by the *Objectname* parameter) and then select the desired relationship with the **-c**, **-s**, or **-t** flag. The information displayed then depends upon the type and class of the anchored object. For example, if you select an object of type **spot**, the type of relationships that can be displayed are:

- Machines that use the Shared Product Object Tree (SPOT) resource.
- Networks that can access the SPOT resource.

When not displaying relationships, the **lsnim** command provides flags that can be used to filter the output that it would normally display. The **-a**, **-c**, **-O**, **-s**, or **-t** flag can be used to restrict the amount of information which is displayed.

Flags

-a *Attribute*	Filters displayed information based on the specified attribute name. The possible attributes are: Operation subclass type class
-c *Class*	Specifies a NIM object class. When this flag is used without the *Objectname* parameter, it filters the displayed information so only information about objects in that class is displayed.
-g	Displays long listing of group object with state information for individual members.
-l	Displays detailed information.
-L	Displays information about resources that can be accessed by a client machine.
-m	Applies other flags specified to group members.
-O	Lists the operations NIM supports.
-p	Displays predefined information using default values.
-P	Displays help information for predefined data.
-q *Operation*	Lists the attributes required for the specified operation.
-S	Displays a list of NIM subclasses.
-s *Subclass*	Specifies a NIM subclass. When this flag is used without the *ObjectName* parameter, it filters the displayed information so only information about objects in that subclass is displayed.
-t *Type*	Specifies a NIM object type. When this flag is used without the *Objectname* parameter, it filters the displayed information so only information about objects of that type is displayed.
-Z	Displays information in colon-separated format.

Security

Access Control: You must have root authority to run the **lsnim** command.

Examples

1. To display a list of NIM object classes, enter:
   ```
   lsnim -p
   ```
2. To display a list of NIM subclasses, enter:
   ```
   lsnim -p -S
   ```
3. To display the list of NIM object types for the `machines` object class, enter:
   ```
   lsnim -p -c machines
   ```
4. To display help information about NIM object types for the `machines` object class, enter:
   ```
   lsnim -P -c machines
   ```
5. To display detailed information about the NIM attributes named `lpp_source` and `Rstate`, enter:
   ```
   lsnim -p -a lpp_source -a Rstate
   ```
6. To display the operations which can be performed on the `paging` object type, enter:
   ```
   lsnim -p -t paging -O
   ```
7. To display the information required to perform a `bos_inst` operation on an object of the `standalone` object type, enter:
   ```
   lsnim -t standalone -q bos_inst
   ```
8. To display information about all customized objects of the `diskless` object type, enter:
   ```
   lsnim -t diskless
   ```
9. To display all customized objects in the `networks` object class, enter:
   ```
   lsnim -c networks
   ```
10. To display detailed information about a NIM object named `altoid`, enter:
    ```
    lsnim -l altoid
    ```
11. To display the relationship between an object named `altoid` and all NIM resources, enter:
    ```
    lsnim -c resources altoid
    ```
12. To display a list of operations that can be applied to `altoid`, enter:
    ```
    lsnim -O altoid
    ```
13. To display a list of resources available to `altoid`, enter:
    ```
    lsnim -L altoid
    ```
14. To display the members of the machine group `MacGrp1` with state and group exclusion status, enter:
    ```
    lsnim -g MacGrp1
    ```
15. To display basic information about the members of the resource group `ResGrp1`, enter:
    ```
    lsnim -m ResGrp1
    ```
16. To display a long listing of members of the machine group `MacGrp1`, with any hidden NIM internal information, enter:
    ```
    lsnim -m -Fl MacGrp1
    ```
17. To display all members of machine group `MacGrp1` which has a spot allocated, enter:
    ```
    lsnim  -ma spot MacGrp1
    ```

Files

/etc/niminfo Contains variables used by NIM.

Related Information

The **nim** command, **nimclient** command, **nimconfig** command, **niminit** command.

The **.info** file.

---

lsnlspath Command

Purpose

Shows the value of the secure NLSPATH system configuration variable.

Syntax

lsnlspath

Description

The **lsnlspath** command outputs the current value of the secure NLSPATH variable.

Related Information

The **chnlspath** command.

---

lsparent Command

Purpose

Displays the possible parent devices that accept a specified connection type or device.

Syntax

lsparent { **-C** | **-P** } { **-k** *ChildConnectionKey* | **-l** *ChildName* } [**-f** *File*] [**-F***Format*] [**-h**] [**-H**]

Description

The **lsparent** command lists devices from the Device Configuration database that can accept a given device as a child device, specified by the **-l** *ChildName* flag, or a given type of child device connection, specified by the **-k** *ChildConnectionKey* flag.

You can display the default output one of two ways. You can either display the default output information for a device from the Customized Devices object class, which is name, state, location, and description, using the **-C** flag, or display the default output information for a device from the Predefined Devices object class, which is class, type, subclass, and description, using the **-P** flag. To override these two default outputs, you can use the **-F** *Format* flag to display the output as designated by a user-formatted string. The *Format* parameter is a quoted list of column names separated and possibly terminated by nonalphanumeric characters.

You can supply the flags either on the command line or from the specified *File* parameter.

Flags

-C	Lists information about a device that is in the Customized Devices object class. The information displayed can be from both the Customized and Predefined Devices object classes. This flag cannot be used with the **-P** flag.
-f *File*	Reads the needed flags from the *File* variable.

-F *Format*	Displays the output in a user-specified format, where the *Format* variable is a quoted list of column names from the Predefined Devices object class or the Customized Devices object class separated and possibly terminated by nonalphanumeric characters. Using white space as the separator, the **lsparent** command displays the output in aligned columns. In addition to the column names in the two object classes, the special name *description* can be used to display a text description of the device.
-H	Displays headers above the column output.
-h	Displays the command usage message.
-k *ChildConnectionKey*	Specifies the connection key that identifies the device subclass name of the child device. This flag cannot be used with the **-l** flag.
-l *ChildName*	Specifies the logical name of a possible child device. This flag cannot be used with the **-k** flag.
-P	Lists information about a device that is in the Predefined Devices object class. The information displayed can be from both the Customized and Predefined Devices object classes. This flag cannot be used with the **-C** flag.

Examples

1. To list possible parent devices in the Customized Devices object class that accept an RS-232 device, type:

   ```
   lsparent -C -k rs232
   ```

   The system displays a message similar to the following:

   ```
   sa0 Available 01-S1 Standard I/O Serial Port
   sa1 Available 01-S2 Standard I/O Serial Port
   sa2 Available 10-68 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa3 Available 10-70 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa4 Available 10-78 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   sa5 Available 20-58 IBM 8-Port EIA-232/RS-422A (PCI) Adapter
   ```

2. To list possible types of parent devices in the Predefined Devices object class that accept an RS-232 device, type:

   ```
   lsparent -P -k rs232
   ```

   The system displays a message similar to the following:

   ```
   adapter      pnp501   isa_sio  Standard I/O Serial Port
   adapter      4f111100 pci      IBM 8-Port EIA-232/RS-422A (PCI)
   Adapter
   concentrator 16c232   sync_pci 16-Port RAN EIA-232 for 128-Port Adapter
   concentrator 16e232   sync_pci 16-Port Enhanced RAN EIA-232 for 128-Port Adapter
   ```

3. To list possible parent devices in the Customized Devices object class that accept the tape device `rmt0` as a child device, type:

   ```
   lsparent -C -l rmt0
   ```

   The system displays a message similar to the following:

   ```
   scsi2 Available 20-60 Wide/Ultra-2 SCSI I/O Controller
   scsi3 Available 20-61 Wide/Ultra-2 SCSI I/O Controller
   scsi1 Available 10-88 Wide/Ultra-2 SCSI I/O Controller
   scsi0 Available 10-60 Wide/Fast-20 SCSI I/O Controller
   ```

4. To list possible types of parent devices in the Predefined Devices object class that accept the tape device `rmt0` as a child device, type:

   ```
   lsparent -P -l rmt0
   ```

   The system displays a message similar to the following:

```
adapter sym896 pci Wide/Ultra-2 SCSI I/O Controller
adapter sym895 pci Wide/Ultra-2 SCSI I/O Controller
adapter sym875 pci Wide/Fast-20 SCSI I/O Controller
```

Files

/usr/sbin/lsparent	Contains the **lsparent** command.

Related Information

The **chdev** command, **lsattr** command, **lsconn** command, **lsdev** command, **mkdev** command, **rmdev** command.

Devices Overview for System Management in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* provides information about adding, changing, moving, and removing devices.

---

lspath Command

Purpose

Displays information about paths to an MultiPath I/O (MPIO) capable device.

Syntax

lspath [**-F** *Format*] [**-H**] [**-l** *Name*] [**-p** *Parent*] [**-s** *Status*] [**-w** *Connection*]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] {**-D** [**-O**] | **-E** [**-O**] | **-F** *Format* [**-Z** *character*] } [**-a** *Attribute*] ...[**-f** *File*] [**-h**] [**-H**]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] { **-D** [**-O**] | **-F** *Format* [**-Z** *character*] } [**-a** *Attribute*] ... [**-f** *File*] [**-h**] [**-H**]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] **-R -a** *Attribute* [**-f** *File*] [**-h**] [**-H**]

lspath -h

Description

The **lspath** command displays one of two types of information about paths to an MPIO capable device. It either displays the operational status for one or more paths to a single device, or it displays one or more attributes for a single path to a single MPIO capable device. The first syntax shown above displays the operational status for one or more paths to a particular MPIO capable device. In this instance, the **lspath** command is similar to the **lsdev** command. The second syntax (keyed by the presence of the **-A** flag) displays one or more attributes for a single path to a particular MPIO capable device. In this instance, the lspath command is similar to the **lsattr** command. In fact, all of the flags for the **lsattr** command are supported on the **lspath** command when displaying path attributes.

Displaying Path Status with the lspath Command

When displaying path status, the set of paths to display is obtained by searching the device configuration database for paths that match the following criteria:

- The target device name matches the device specified with the **-l** flag. If the **-l** flag is not present, then the target device is not used in the criteria.
- The parent device name matches the device specified with the **-p** flag. If the **-p** flag is not present, then parent is not used in the criteria.

- The connection matches the connection specified with the **-w** flag. If the **-w** flag is not present, then connection is not used in the criteria.
- The path status matches status specified with the **-s** flag. If the **-s** flag is not present, the path status is not used in the criteria.

If none of the **-l**, **-p**, **-w**, **-s** flags are specified, then all paths known to the system are displayed.

By default, this command will display the information in columnar form. When no flags are specified that qualify the paths to display, the format of the output is:

```
status device  parent
```

The default display format can be overridden by using the **-F** *Format* flag. The **-F** *Format* flag displays the output in a user-specified format where the *Format* parameter is a quoted list of field names separated by, and possibly ended by, non-alphanumeric characters or white space. The field names are the fields defined in the **CuPath** class or one of the column heading defined above.

Note: The column names above are not translated into other languages (either when output as column headings or when input as part of the *Format* of the **-F** flag).

Possible values that can appear for the status column are:

enabled
Indicates that the path is configured and operational. It will be considered when paths are selected for IO.

disabled
Indicates that the path is configured, but not currently operational. It has been manually disabled and will not be considered when paths are selected for IO.

failed Indicates that the path is configured, but it has had IO failures that have rendered it unusable. It will not be considered when paths are selected for IO.

defined
Indicates that the path has not been configured into the device driver.

missing
Indicates that the path was defined in a previous boot, but it was not detected in the most recent boot of the system.

detected
Indicates that the path was detected in the most recent boot of the system, but for some reason it was not configured. A path should only have this status during boot and so this status should never appear as a result of the **lspath** command.

Displaying Path Attributes with the lspath Command

When displaying attributes for a path, the path must be fully qualified. Multiple attributes for a path can be displayed, but attributes belonging to multiple paths cannot be displayed in a single invocation of the **lspath** command. Therefore, in addition to the **-A** flag, the **-l**, **-p**, or**-w** flags are required to uniquely identify a single path. For example:

- if only one path exists to a device, the **-l** flag is required
- if only one path between a device and a specific parent, the **-l** and **-p** flags are required
- if there are multiple paths between a device and a specific parent, the **-l**, **-p**, and **-w** flags are required

Furthermore, the **-s** flag is not allowed.

The same rules used by the **lsattr** command for displaying device attributes applies to the **lspath** command for displaying path attributes.

By default, this command will display the information in columnar form. The format of the output is the same as the **lsattr** command:

```
attribute   value    description         user_settable
```

All fields are shown by default. The default display format can be overridden by using the **-F** *Format* flag. The **-F** *Format* flag displays the output in a user-specified format where the *Format* parameter is a quoted list of column names separated by, and possibly ended by, non-alphanumeric characters or white space. The column names allowed are the field names from the **CuPathAt**, **PdPathAt**, and **PdAtXtd** object classes plus the columns listed above. The **PdAtXtd** based information is used by the Web-based System Manager.

Note: The column names above are not translated into other languages (either when output as column headings or when input as part of the *Format* of the **-F** flag).

Flags

-a *Attribute*
Identifies the specific attribute to list. The '*Attribute*' is the name of a path specific attribute. When this flag is provided, only the identified attribute is displayed. Multiple instances of this flag may be used to list multiple attributes. If this flag is not specified at all, all attributes associated with the identified path will be listed.

-A
Indicates that attributes for a specific path are to be displayed. When the **-A** flag is present, the **-s** *Status* flag is not allowed. However, the **-l** *Name*, **-p** *Parent*, and **-w** *Connection* flags must be present to fully qualify the path.

-D
Displays the attribute names, default values, descriptions, and user-settable flag values for a specific path when not used with the **-O** flag. The **-D** flag displays only the attribute name and default value in colon format when used with the **-O** flag. This flag is only valid when displaying path attributes and it cannot be used with the **-E**, **-F**, or **-R** flag.

-E
Displays the attribute names, current values, descriptions, and user-settable flag values for a specific path when not used with the **-O** flag. The **-E** flag displays only the attribute name and current value in colon format when used with the **-O** flag. This flag is only valid when displaying path attributes and it cannot be used with the **-D**, **-F**, or **-R** flag.

-f *File*
Reads the needed flags from the *File* parameter.

-F *Format*
Displays the output in a user-specified format, where the *Format* parameter is a quoted list of column names separated by non-alphanumeric characters or white space. Using white space as the separator, the lspath command displays the output in aligned columns. Valid column names depends upon the type of information requested. For path display, column names from the **CuPath** object class can be specified. For path attribute display (the **-A** flag is specified), column names from the **PdPathAt** and **CuPathAt** object classes can be specified. In addition to the column names, there are two special purpose names that can be used. The name description can be used to obtain a display of attribute descriptions and user-settable can be used to obtain an indication as to whether or not an attribute can be changed. This flag cannot be used with the **-E**, **-D**, **-O** or **-R** flag.

-h
Displays the command usage message.

-H Displays headers above the column output. To use the **-H** flag with the **-O** flag is meaningless, the **-O** flag prevails. To use the **-H** flag with the **-R** flag is meaningless; the **-R** flag prevails.

-l *Name* Specifies the logical device name of the target device whose path information is to be displayed. This flag is optional for displaying path status, but is required for displaying path attributes.

-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 is only valid when displaying path attributes and it cannot be used with the **-F** and **-R** flags.

-p *Parent* Indicates the logical device name of the parent device of the path(s) to be displayed. This flag is optional for displaying path status, but is required for displaying path attributes.

-R Displays the legal values for an attribute name. The **-R** flag cannot be used with the **-D**, **-E**, **-F** and **-O** flags. The **-R** flag displays the list attribute values in a vertical column as follows:

```
Value1
Value2
.
.
ValueN
```

The **-R** flag displays the range attribute values as x...n(+i) where x is the start of the range, n is the end of the range, and i is the increment.

-s *Status* The **-s** *Status* flag indicates the status to use in qualifying the paths to be displayed. When displaying path information, the allowable values for this flag are:

enabled
Display paths that are **enabled** for MPIO path selection.

disabled
Display paths that are **disabled** from MPIO path selection.

failed Display paths that are **failed** due to IO errors.

available
Display paths whose **path_status** is **PATH_AVAILABLE** (that is, paths that are configured in the system, includes **enabled**, **disabled**, and **failed** paths).

defined
Display paths whose **path_status** is **PATH_DEFINED**.

missing
Display paths whose **path_status** is **PATH_MISSING**.

-w *Connection* Indicates the connection information to use in qualifying the paths to be displayed. This flag is optional for displaying path status, but is required for displaying path attributes.

-Z *Character*

The **-Z** *Character* flag is meant to be used for programs that need to deal with ODM fields that may have embbeded new line characters. An example of such a program is the Web-based System Manager. The **-Z** *Character* flag is used to change the record separator character for each record (line) of output generated. The new record separator is the *'Character'* argument to this flag. The **-Z** *Character* flag is only relevant when the **-A** and the **-F** *Format* flags are specified. The **-Z** *Character* flag cannot be used with the **-D**, **-E**, **-O**, or the **-R** flags.

Security

Privilege Control: All users can execute this command.

Auditing Events: None.

Examples

Examples of displaying path status:

1. To display the status of all paths to hdisk1 with column headers, enter:

   ```
   lspath -H -l hdisk1
   ```

   The system will display a message similar to the following:

   ```
   status    device   parent
   enabled   hdisk1   scsi0
   disabled  hdisk1   scsi1
   missing   hdisk1   scsi2
   ```

2. To display, without column headers, the set of paths whose operational status is disabled, enter:

   ```
   lspath -s disabled
   ```

   The system will display a message similar to the following:

   ```
   disabled  hdisk1   scsi1
   disabled  hdisk2   scsi1
   disabled  hdisk23  scsi8
   disabled  hdisk25  scsi8
   ```

3. To display the set of paths whose operational status is `failed`, enter:

   ```
   lspath -s failed
   ```

   The system will display a message similar to the following:

   ```
   failed  hdisk1   scsi1
   failed  hdisk2   scsi1
   failed  hdisk23  scsi8
   failed  hdisk25  scsi8
   ```

4. To display in a user-specified format, without column headers, the set of paths to hdisk1 whose path status is available enter:

   ```
   lspath -l hdisk1 -s available -F"connection:parent:path_status:status"
   ```

   The system will display a message similar to the following:

   ```
   5,0:scsi0:available:enabled
   6,0:scsi1:available:disabled
   ```

   Note that this output shows both the path status and the operational status of the device. The path status simply indicates whether the path is configured or not. The operational status indicates how the path is being used with respect to path selection processing in the device driver. Only paths with a

path status of `available` also have an operational status. If a path is not currently configured into the device driver, it does not have an operational status.

Examples of displaying path attributes:

1. If the target device is a SCSI disk, to display all attributes for the path to parent scsi0 at connection 5,0, use the command:

   ```
   lspath -AHE -l hdisk10 -p scsi0 -w "5,0"
   ```

   The system will display a message similar to the following:

   ```
   attribute  value  description                      user_settable
   weight     1      Order of path failover selection  true
   ```

Related Information

The **chpath** command, **mkpath** command, **rmpath** command.

---

lsprtsv Command

Purpose

Shows print service information stored in the database.

Syntax

lsprtsv { **-c** | **-p** } [**-h**] [**-q***QEntry ...*] [**-Z**]

Description

The **lsprtsv** high-level command shows predefined and customized TCP/IP-based print service information. Use the **lsprtsv** command to extract customized or predefined print service information.

The **lsprtsv** command can show the following information:

- A list of host names that have access rights to the print server
- Logical print queue information only

Flags

-c	Extracts customized configuration information.
-h	Shows a list of host names that can use the print server.
-p	Extracts predefined configuration information.
-q*QEntry...*	Shows the logical print queues specified and their attributes available on a host. The *QEntry* variable specifies the names of the queues to display.
-Z	Specifies that the output be produced in colon format. This flag is used if the **lsprtsv** command is invoked from the SMIT usability interface.

Examples

1. To show all host names who have access rights to a print server, enter:

   ```
   $ lsprtsv  -c  -h
   ```

2. To show which logical printers are available on a given client machine, enter:

   ```
   lsprtsv  -c  -q sahara
   ```

Related Information

The **chque** command, **chquedev** command, **ruser** command.

The **lpd** daemon, **qdaemon** daemon.

Understanding the SMIT Interface for TCP/IP in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

lsps Command

Purpose

Displays the characteristics of a paging space.

Syntax

lsps { **-s** | [**-c** | **-l**] { **-a** | **-t** { *lv* | *nfs* } | *PagingSpace* } }

Description

The **lsps** command displays the characteristics of a paging space. The **lsps** command displays characteristics such as the paging-space name, physical-volume name, volume-group name, size, percentage of the paging space used, whether the space is active or inactive, and whether the paging space is set to automatic. The *PagingSpace* parameter specifies the paging space whose characteristics are to be shown.

For NFS paging spaces, the physical-volume name and volume-group name will be replaced by the host name of the NFS server and the path name of the file that is used for paging.

You can use the File Systems application in Web-based System Manager (wsm) to change file system characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsps** fast path to run this command.

Flags

-a Specifies that the characteristics of all paging spaces are to be given. The size is given in megabytes.

-c Specifies that the output should be in colon format. The colon format gives the paging space size in logical partitions.

-l Specifies that the output should be in list format.

-s Specifies that the summary characteristics of all paging spaces are to be given. This information consists of the total paging space in megabytes and the percentage of paging space currently assigned (used). If the **-s** flag is specified, all other flags are ignored.
Note: Setting the environment variable **PSALLOC=early** causes the use of early paging space algorithm. In this case, the value the **-s** flag specifies is different from the value returned for a single paging space or when using the **-a** flag for all the paging spaces. The value the **-s** flag displays is the percentage of paging space allocated (reserved), whether the paging space has been assigned (used) or not. Therefore, the percentage reported by the **-s** flag is usually larger than that reported by the **-a** flag when **PSALLOC** is set to early.

-t Specifies the characteristics of the paging space. One of the following variables is required:

lv Specifies that the characteristics of only logical volume paging spaces are to be given.

nfs Specifies that the characteristics of only NFS paging spaces are to be given. The heading of the output will be changed to display the host name of the NFS server and the path name of the file that resides on the server that is being used for NFS paging.

Examples

To list the characteristics of all paging spaces, enter:

```
lsps  -a
```

This displays the characteristics for all paging spaces and provides a listing similar to the following listing:

```
Page Space      Physical Volume   Volume Group    Size %Used Active  Auto  Type
hd6             hdisk0            rootvg         512MB     1     yes   yes    lv
```

Files

/etc/swapspaces	Specifies the paging space devices activated by the **swapon -a** command.

Related Information

The **mkps** command, **chps** command, **rmps** command, **swapon** command.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

Paging Space Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains paging space and its allocation policies.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the structure, main menus, and tasks that are done with SMIT.

---

lspv Command

Purpose

Displays information about a physical volume within a volume group.

Syntax

lspv

OR

lspv [**-L**] [**-l** | **-p** | **-M**] [**-n** *DescriptorPhysicalVolume*] [**-v** *VolumeGroupID*] *PhysicalVolume*

Description

The **lspv** command displays information about the physical volume if the specific physical volume name is specified. If you do not add flags to the **lspv** command, the default is to print every known physical volume in the system along with its physical disk name, physical volume identifiers (PVIDs), to which volume group (if any) the physical volume belongs, and the state of the volume group as active if the volume group is varied on or concurrent if it is varied on in concurrent mode.

Note: If the **lspv** command cannot find information for a field in the Device Configuration Database, it will insert a question mark (?) in the value field. As an example, if there is no information for the `PP RANGE` field, the following might be displayed:

```
PP RANGE: ?
```

The **lspv** command attempts to obtain as much information as possible from the description area when it is given a logical volume identifier.

When the *PhysicalVolume* parameter is used, the following characteristics of the specified physical volume are displayed:

`Physical volume`	Name of the physical volume.
`Volume group`	Name of volume group. Volume group names must be unique systemwide names and can be from 1 to 15 characters long.
`PV Identifier`	The physical volume identifier for this physical disk.
`VG Identifier`	The volume group identifier of which this physical disk is a member.
`PVstate`	State of the physical volume. If the volume group that contains the physical volume is varied on with the **varyonvg** command, the state is `active`, `missing`, or `removed`. If the physical volume is varied off with the **varyoffvg** command, the state is `varied off`.
`Allocatable`	Allocation permission for this physical volume.
`Logical volumes`	Number of logical volumes using the physical volume.
`Stale PPs`	Number of physical partitions on the physical volume that are not current.
`VG descriptors`	Number of volume group descriptors on the physical volume.
`PP size`	Size of physical partitions on the volume.
`Total PPs`	Total number of physical partitions on the physical volume.
`Free PPs`	Number of free physical partitions on the physical volume.
`Used PPs`	Number of used physical partitions on the physical volume.
`Free distribution`	Number of free partitions available in each intra-physical volume section.
`Used distribution`	Number of used partitions in each intra-physical volume section.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit lspv** fast path to run this command.

Flags

-L Specifies no waiting to obtain a lock on the Volume group. **Note:** If the volume group is being changed, using the **-L** flag gives unreliable date.

-l Lists the following fields for each logical volume on the physical volume:

LVname Name of the logical volume to which the physical partitions are allocated.

LPs The number of logical partitions within the logical volume that are contained on this physical volume.

PPs The number of physical partitions within the logical volume that are contained on this physical volume.

Distribution
The number of physical partitions, belonging to the logical volume, that are allocated within each of the following sections of the physical volume: outer edge, outer middle, center, inner middle and inner edge of the physical volume.

Mount Point
File system mount point for the logical volume, if applicable.

-M Lists the following fields for each logical volume on the physical volume:

```
PVname:PPnum [LVname: LPnum [:Copynum] [PPstate]]
```

Where:

PVname Name of the physical volume as specified by the system.

PPnum Physical partition number.

LVname Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters.

LPnum Logical partition number. Logical partition numbers can range from 1 to 64,000.

Copynum
Mirror number.

PPstate
Only the physical partitions on the physical volume that are not current are shown as stale.

-n *DescriptorPhysicalVolume* Accesses information from the variable descriptor area specified by the *DescriptorPhysicalVolume* variable. The information may not be current, since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the validated information is accessed, and therefore the information displayed is current. The volume group need not be active when you use this flag.

-p Lists the following fields for each physical partition on the physical volume:

Range A range of consecutive physical partitions contained on a single region of the physical volume.

State The current state of the physical partitions: `free`, `used`, `stale`, or `vgda`.
Note: If a *volume group* is converted to a big vg format, it may be necessary to use some data partitions for *volume group* descriptor area. These partitions will be marked `vgda`.

Region The intra-physical volume region in which the partitions are located.

LVname The name of the logical volume to which the physical partitions are allocated.

Type The type of the logical volume to which the partitions are allocated.

Mount point
File system mount point for the logical volume, if applicable.

-v *VolumeGroupID* Accesses information based on the *VolumeGroupID* variable. This flag is needed only when the lspv command does not function due to incorrect information in the Device Configuration Database. The *VolumeGroupID* variable is the hexadecimal representation of the volume group identifier, which is generated by the **mkvg** command.

Examples

1. To display the status and characteristics of physical volume `hdisk3`, enter:

   ```
   lspv hdisk3
   ```

2. To display the status and characteristics of physical volume `hdisk5` by physical partition number, enter:

   ```
   lspv  -p hdisk5
   ```

3. To display the status and characteristics of physical volume `hdisk5` using the volume group ID, enter:

   ```
   lspv  -v 00014A782B12655F hdisk5
   ```

   The following is an example of the output:

   ```
   lspv
   hdisk0   0000000012345678  rootvg   active
   hdisk1   10000BC876543258  vg00     concurrent
   hdisk2   ABCD000054C23486  None
   ```

   The previous example shows that physical volume hdisk0 contains the volume group rootvg, and it is varied on. Physical volume hdisk1 contains the volume group vg00, and it is varied on in concurrent mode. Physical volume hdisk2 does not contain a varied on volume group.

Files

/usr/sbin Contains the **lspv** command.

Related Information

The **chpv** command, **lslv** command, **lsvg** command**, mklv** command, **varyonvg** command.

Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the Logical Volume Manager, physical volumes, logical volumes, volume groups, organization, ensuring data integrity, and allocation characteristics.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the structure, main menus, and tasks that are done with SMIT.

---

lsque Command

Purpose

Displays the queue stanza name.

Syntax

lsque [**-c**] **-q***Name*

Description

The **lsque** command uses the **printf** subroutine to display the name of the queue stanza and associated attributes from the **/etc/qconfig** file.

Flags

-c	Causes colon output format for use by SMIT.
-q *Name*	Specifies the *Name* of the queue stanza that is sent to standard output.

Examples

1. To display the name of queue stanza `lp0`, enter:

   ```
   lsque  -qlp0
   ```

   A list similar to the following is displayed:

   ```
   lp0:
      device = lpd0
      host = neptune
      rq = nlp0
   ```

2. To display the name of queue stanza `lp0` in colon format, enter:

   ```
   lsque  -c  -q lp0
   ```

   A list similar to the following is displayed:

   ```
   device:discipline:up:acctfile:host:s_statfilter:l_statfilter:rq
   lpd0:fcfs:true:false:neptune:::nlp0
   ```

Files

/usr/bin/lsque	Contains the **lsque** command.
/etc/qconfig	Contains the configuration file.

Related Information

The **chque** command, **lsquedev** command, **mkque** command, **rmque** command.

The **qconfig** file.

The **printf** subroutine.

Printer Specific Information, Printer Support, and Virtual Printer Definitions and Attribute Values in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

lsquedev Command

Purpose

Displays the device stanza name.

Syntax

lsquedev [**-c**] **-q***Name* **-d** *Name*

Description

The **lsquedev** command displays the name of the queue stanza and associated attributes from the **/etc/qconfig** file.

Flags

-c	Specifies colon output format for use by SMIT.
-d *Name*	Specifies the *Name* variable of the device stanza that is displayed.
-q *Name*	Specifies the *Name* variable of the queue containing the device stanza that is displayed.

Examples

1. To display device stanza `dlp0` on the `lp0` queue, type:

   ```
   lsquedev  -q lp0  -d dlp0
   ```

   A listing similar to the following is displayed:

   ```
   dlp0:
   FILE = /dev/lp0
   BACKEND = /usr/lib/lpd/piobe
   ```

2. To display device stanza `dlp0` on the `lp0` queue in colon format, type:

   ```
   lsquedev  -c  -qlp0  -d dlp0
   ```

   A listing similar to the following is displayed:

   ```
   file:access:feed:header:trailer:backend:align
   dlp0:/dev/lp0:read:never:never:never:/usr/lib/lpd/piobe:TRUE
   ```

Files

/usr/bin/lsquedev	Contains the **lsquedev** command.
/etc/qconfig	Contains the configuration file.

Related Information

The **chquedev** command, **lsque** command, **mkquedev** command, **rmquedev** command.

The **qconfig** file.

The **printf** subroutine.

Printer Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Specific Information, Printer Support, and Virtual Printer Definitions and Attribute Values in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

lsresource Command

Purpose

Displays bus resources for available devices in the system and recommends attribute values for bus resource resolution.

Syntax

lsresource [**-a** | **-r**] [**-d**] **-l** *Name*

Description

The **lsresource** command writes a list of assigned bus resources to standard out, or determines if the bus resources for devices resolve.

The **lsresource** command lets you display the currently assigned values for the bus resource attributes for the device specified by the given device logical name (**-l** *Name*). Use the **-a** flag to display the currently assigned values for all bus resource attributes for all devices under the same parent bus as the specified device.

Use the **-r** flag to determine if the bus resources for the specified device are resolvable. In this case, the **lsresource** command checks all devices under the same parent bus as the specified device, including defined but not available devices, to see if their bus resource attributes are resolvable. The **lsresource** command produces no output if all attributes resolve. The **lsresource** command provides information depending on the type of conflict detected if any device's bus resources are unresolvable. In some cases, the **lsresource** command can provide you with information that leads to the resolution of the conflict.

The **lsresource** command identifies the device name, attribute name, and a suggested value for the attribute if a conflict results from an attribute that only a user can change. Setting the attribute to the suggested value should resolve the conflict. This may aid in the configuration of devices with attributes that can only a user can change. Such devices include adapter cards which use jumpers or switches on the card to select values.

In some cases, a conflict may be due to an attribute which the system can normally adjust at boot time but is prevented from doing so at run time because the device is in the Available state. In these situations, the **lsresource** command will indicate that the configuration will be resolved by rebooting the system.

It is possible that multiple user changeable attributes will be identified when unresolvable conflicts occur. These may be for the device specified by the given device logical name (**-l** *Name*) or for other devices in the system. All of the identified attributes will need to be changed to resolve the conflict. It may even be the case where user changeable attributes are identified and a reboot is indicated. In this case, all of the identified attributes will need to be changed and the system rebooted to resolve the conflicts.

Finally, **lsresource** may determine that the set of devices currently defined in the devices configuration database can not be resolved regardless of attributes being changed or the system rebooted. In this case, a list of the devices which could not be resolved is written to standard out. If the problem has resulted from a new device just being defined, that device should be removed, or the devices listed by lsresource should be removed. If the problem is not resolved by removing devices, there could be additional problems on the next reboot. This is because the order in which devices are resolved at boot time may differ from the order they are resolved by lsresource, resulting in a different set of unresolvable devices at boot time. If the set of unresolvable devices at boot time should now include a device needed for booting, problems such as no console being configured or the system failing to boot could occur.

The following applies when lsresource is used to list currently assigned bus resource values (the **-r** flag is not specified).

The **TYPE** field in the output listing contains the following symbols:

B Bus Memory Address Values
M Bus Memory Address Values
O I/O Address Values
I Bus Interrupt Levels
N Non-sharable Bus Interrupt Levels
A DMA Arbitration Level

The **S** column denotes shared attributes. These are attributes which are required to be set to the same value. They are grouped by the number specified in the column. All attributes with a `1` by them must be set to the same value, all attributes with a `2` by them must be set to the same value, and so on. In some cases, two or more interrupt attributes may be set to the same value but have no numbers in the **S** column indicating that they are shared. This is because the values are not required to be the same but just happen to be set to the same value because they could not be assigned their own unique values.

The **G** column denotes attributes in a group. These are a set of attributes whose values depend on each other. If one is changed to the next possible value, the rest of the attributes in the group must also be changed to the next possible value. Their groupings are indicated by the number specified in the column. All attributes with a `1` by them are in the same group, all attributes with a `2` by them are same group, and so on.

On some models, the interrupt value displayed may be followed by a value enclosed in parenthesis. This is not part of the interrupt value but serves to identify the interrupt controller to which the interrupt is associated. The identifier consists of a letter followed by a number, such as `A0`. The letter indicates the type of interrupt controller and the number distinguishes between multiple instances of that type of controller. There are two types of interrupt controllers that may be identified:

A Indicates an AT interrupt controller.
B Indicates a non-AT interrupt controller.

Flags

-a Specifies that all allocated bus resource attributes for all devices connected to the same top parent bus as the device specified with the **-l** flag are to be displayed. This flag cannot be used with the **-r** flag.
-d Specifies that the attribute text descriptions are to be included in the output.
-l *Name* (Lowercase L) Specifies the logical name of the device attributes to display.
-r Specifies to attempt to resolve all bus resources of all devices connected to the same top parent bus as the device specified with the **-l** flag. This will include all devices that are in the DEFINED state. The **lsresource** command will display any conflicts and advise the user on changeable values. No changes to the ODM database are made. This flag cannot be used with the **-a** flag.

Security

Access Control: Any User

Auditing Events: N/A

Examples

1. To list bus attributes for the token ring device, enter:

```
lsresource -l tok0
```

The system displays a message similar to the following:

```
TYPE   DEVICE         ATTRIBUTE       S G CURRENT VALUE
----- -------------  --------------- - - ---------------------
M     tok0           dma_bus_mem         0x003b2000 - 0x003f1fff
O     tok0           bus_io_addr         0x000086a0 - 0x000086af
N     tok0           bus_intr_lvl                3
A     tok0           dma_lvl                     7
```

2. To list bus attributes for all devices, enter:

```
lsresource -a -l tok0
```

The system displays a message similar to the following:

```
TYPE  DEVICE          ATTRIBUTE          S G   CURRENT VALUE
----- -------------  ---------------    - -    ----------------------
M      bus0           bus_iocc_mem             0x00fffff0 - 0x00ffffff
M      gda0           vram_start         1     0x00400000 - 0x007fffff
M      gda0           bus_mem_start            0x000c0000 - 0x000c1fff
M      gda0           dma1_start               0x00800000 - 0x009fffff
M      gda0           dma2_start               0x00a00000 - 0x00bfffff
M      gda0           dma3_start               0x00c00000 - 0x00dfffff
M      gda0           dma4_start               0x01000000 - 0x011fffff
M      scsi0          bus_mem_addr             0x000e0000 - 0x000e0fff
M      scsi0          dma_bus_mem              0x00100000 - 0x00301fff
M      tok0           dma_bus_mem              0x003b2000 - 0x003f1fff
O      da0            bus_io_addr              0x00000060 - 0x0000006f
O      siokta0        bus_io_addr              0x00000050 - 0x00000051
O      sioma0         bus_io_addr              0x00000048 - 0x00000049
O      ppa0           bus_io_addr              0x00000078 - 0x0000007a
O      gda0           bus_addr_start     1     0x00002110 - 0x0000211f
O      tok0           bus_io_addr              0x000086a0 - 0x000086af
I      siokta0        bus_intr_lvl                        1   (A0)
I      sioma0         bus_intr_lvl                        1   (A0)
I      ppa0           bus_intr_lvl                       13   (A0)
I      gda0           int_level                           9   (A0)
I      scsi0          bus_intr_lvl                       14   (A0)
N      fda0           bus_intr_lvl                        6   (A0)
N      tok0           bus_intr_lvl                        3   (A0)
A      fda0           dma_lvl                             0
A      gda0           dma_channel                         3
A      scsi0          dma_lvl                             4
A      tok0           dma_lvl                             7
```

3. To report the outcome of a resolution of device attributes, enter:

```
lsresource -r - d -l tok0
```

Depending on the outcome of the resolution, different messages may be displayed. The output below signifies to a user that the resolution can be successful if changes are made, i.e., the attributes are changed to the suggested values.

```
lsresource: The attribute(s) for some device(s) in the system could
not be resolved.  To resolve conflicts, attribute(s) need to be
modified.  A suggested value for each attribute is provided.

DEVICE          ATTRIBUTE        CURRENT   SUGGESTED  DESCRIPTION
--------------  ---------------  -------   ---------  -----------
ent1            bus_intr_lvl     11        5          Bus interrupt level
ent1            bus_mem_addr     0xc0000   0xc4000    Bus memory address
ent1            bus_io_addr      0x300     0x320      Bus I/O address
ent2            bus_intr_lvl     11        7          Bus interrupt level
ent2            bus_mem_addr     0xc0000   0xc8000    Bus memory address
```

Files

/usr/sbin/lsresource	Contains the **lsresource** command.

lsresponse Command

Purpose

Lists information about one or more responses.

Syntax

lsresponse [**–C** | **–l** | **–t** | **–d** | **–D** *delimiter*] [**–A**] [**–q**] [**–x**] [**–a**] [**–h**] [**–TV**] [*response1*[**,***response2***,**...] **:***node_name*]

Description

The **lsresponse** command lists the following information about defined responses:

Name Specifies the name of the response.

Location Specifies the location of the response.

Action Specifies the name of an action.

DaysOfWeek

Specifies the days of the week when the action can be run. **DaysOfWeek** and **TimeOfDay** together define the interval when the action can be run.

The values for the days can be separated by plus signs (**+**) or displayed as a range of days separated by a hyphen (**-**). Multiple **DaysOfWeek** parameters are separated by commas (**,**). The number of **DaysOfWeek** parameters matches the number of **TimeOfDay** parameters. The values for each day follow:

1 Sunday
2 Monday
3 Tuesday
4 Wednesday
5 Thursday
6 Friday
7 Saturday

TimeOfDay

Specifies the time range when **Action** can be run, consisting of the start time followed by the end time separated by a hyphen. **DaysOfWeek** and **TimeOfDay** together define the interval when the action can be run.

The time is in 24-hour format (HHMM), where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. Multiple **TimeOfDay** parameters are separated by commas (**,**). The number of **DaysOfWeek** parameters matches the number of **TimeOfDay** parameters.

ActionScript Specifies the script or command to run for the action.

ReturnCode Specifies the expected return code for **ActionScript**.

CheckReturnCode
Specifies whether the actual return code for **ActionScript** is compared to its expected return code. The valid values are: **y** (yes) and **n** (no).

EventType Specifies the type of event that causes the action to be run: event, rearm event, or both.

StdOut Specifies whether standard output is directed to the audit log. The valid values are: **y** (yes) and **n** (no).

EnvVars Specifies any environment variables that will be set before the action is run. standard output is directed to the audit log. The valid values are: **y** (yes) and **n** (no).

UndefRes Specifies whether the action is to be run if a monitored resource becomes undefined. The valid values are: **y** (yes) and **n** (no).

To get a list of all response names, run the **lsresponse** command alone without any response names specified. A list of all response names is returned. The default format in this case is tabular.

Specifying a node name after the response names limits the display to the responses defined on that node. List all of the responses on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the responses are listed. For local scope, only responses on the local node are listed. Otherwise, the responses from all nodes within the domain are listed.

To see all the information about all response names, specify the **-A** flag with the **lsresponse** command. The **-A** flag causes all information about a response to be listed when no response names are specified. When all of the information about all responses is listed, the long format is the default.

When more than one response is specified, the response information is listed in the order in which the responses are entered.

Flags

–a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lsresponse -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–C Displays the **mkresponse** command that can be used to create the response and one of its actions. If more than one response is specified, each **mkresponse** command appears on a separate line. This flag is ignored when no responses are specified. This flag overrides the **–l** flag.

–l Displays the response information on separate lines (long form).

–t Displays the response information in separate columns (table form).

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **–D** flag if you wish to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

–A Displays all of the attributes of the response.

–q Does not return an error when **response** does not exist.

–x Suppresses headers when printing.

–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

*response1***[,***response2***,...]**
This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring is listed.

node_name Specifies the node where the response 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.

Security

The user needs read permission for the **IBM.EventResponse** resource class to run **lsresponse**. 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.

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

1. To list all of the responses, run this command:

   ```
   lsresponse
   ```

   The output will look like this:

   ```
   Name
   "E-mail root any time"
   "E-mail root first shift"
   "Critical notifications"
   "Generate SNMP trap"
   ```

2. To list general information about the response ″Critical notifications″, run this command:

   ```
   lsresponse "Critical notifications"
   ```

   The output will look like this:

   ```
   Name             = "Critical notifications"
   Location         = "nodeA"
   Action           = "Log Critical Event"
   DaysOfWeek       = 1+2+7
   TimeOfDay        = 0000-2400
   ActionScript     = "/usr/sbin/rsct/bin/logevent /tmp/criticalEvents"
   ReturnCode       = 0
   CheckReturnCode  = "y"
   EventType        = "b"
   StdOut           = "y"
   EnvVars          = '"Env1=5","Env=10"'
   UndefRes         = "n"

   Name             = "Critical notifications"
   Location         = "nodeA"
   Action           = "E-mail root"
   DaysOfWeek       = 6+2,6+2,6+5
   TimeOfDay        = 1700-2400,0000-0800,0000-2400
   ActionScript     = "/usr/sbin/rsct/bin/notifyevent root"
   ReturnCode       = 0
   CheckReturnCode  = "y"
   EventType        = "b"
   StdOut           = "y"
   EnvVars          = ""
   UndefRes         = "n"
   ```

3. To list the command that would create the response ″Critical notifications″ along with one of its actions, run this command:

   ```
   lsresponse -C "Critical notifications"
   ```

   The output will look like this:

   ```
   mkresponse -n "Log Critical Event" -d 1+2+7 -t 0000-2400 \
   -s "usr/sbin/rsct/bin/logevent /tmp/criticalEvents"  \
   -e b -r 0 "Critical notifications"
   ```

4. To list all responses that have the string **E-mail** in their names, run this command:

   ```
   lsresponse "E-mail"
   ```

   The output will look like this:

```
Name   = "E-mail root any time"
Action = "E-mail root"
:
Name   = "E-mail root first shift"
Action = "E-mail root"
```

Location

/usr/sbin/rsct/bin/lsresponse Contains the **lsresponse** command

Related Information

ERRM commands: **chresponse**, **lscondresp**, **mkcondresp**, **mkresponse**, **rmresponse**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations

---

lsrole Command

Purpose

Displays role attributes. This command applies only to AIX 4.2.1 and later.

Syntax

lsrole [**-c** | **-f**] [**-a** *List*] { **ALL** | *Name* [,*Name*] *... }*

Description

The **lsrole** command displays the role attributes. You can use this command to list all attributes of all the roles or all the attributes of specific roles. Since there is no default parameter, you must enter the **ALL** keyword to see the attributes of all the roles. By default, the **lsrole** command displays all role attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsrole** command lists as much information as possible.

By default, the **lsrole** command lists each role's attributes on one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the role attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-c** flag.

You can use the Users application in Web-based System Manager (wsm) to change user characteristics. You could also use the System Management Interface Tool (SMIT) to run this command.

Flags

-a *List* Lists the attributes to display. The *List* variable can include any attribute defined in the **chrole** command and requires a blank space between attributes. If you specify an empty list, only the role names are displayed.

-c Displays the role attributes in colon-separated records, as follows:

```
# role:  attribute1:  attribute2:  ...
  Role:  value1:      value2:      ...
```

-f Displays the output in stanzas, with each stanza identified by a role name. Each *Attribute*=*Value* pair is listed on a separate line:

```
Role:
       attribute1=value
       attribute2=value
       attribute3=value
```

Security

Files Accessed:

Mode	File
r	/etc/security/roles

Examples

To display the role `rolelist` and groups of the role **ManageAllUsers** in colon format, enter:

```
lsrole -c -a rolelist groups ManageAllUsers
```

Information similar to the following appears:

```
# role: rolelist:groups
 ManageAllUsers: ManagerBasicUser:security
```

Files

/etc/security/roles	Contains the attributes of roles.

Related Information

The **chrole** command, **chuser** command, **lsuser** command, **mkrole** command, **mkuser** command, **rmrole** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

lsrpdomain Command

Purpose

Displays peer domain information for the node.

Syntax

lsrpdomain [**–o** | **–O**] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] [*domain_name*]

Description

The **lsrpdomain** command displays information about the peer domains that the node where the command runs belongs to. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify a domain name, the command displays information about that peer domain only. The **-o** and **-O** flags also limit the information this command displays. The **-o** flag displays information only about the online peer domain. The **-O** flag displays information only about peer domains that are offline.

By default, the **lsrpdomain** command displays information in table format (**-t**).

Some of the peer domain information that is displayed follows:

Name Specifies the name of the peer domain.

RSCTActiveVersion
Specifies the version of RSCT that is active in the peer domain.

MixedVersions
Indicates whether more than one version of RSCT is active in the peer domain.

TSPort
Specifies the topology services port number.

GSPort
Specifies the group services port number.

OpState
Specifies the current state of the peer domain.

Flags

-o Displays information about the node's online peer domain.

-O Displays information about peer domains that are offline for the node.

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default format.

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x Excludes the header (suppresses header printing).

-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

domain_name Specifies one or more peer domains to display information about. You can specify a domain name or a substring of a domain name for this parameter. If you specify a substring, the command displays information about any defined domain with a name that contains the substring.

Security

The user of the **lsrpdomain** command needs read permission for the **IBM.PeerDomain** resource class on the node on which the command runs. 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 the node for which the peer domain information is requested.

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

1. To display general information about the peer domains that **nodeA** belongs to, run this command on **nodeA**:

   ```
   lsrpdomain
   ```

   The output will look like this:

   ```
   Name  OpState  RSCTActiveVersion MixedVersions TSPort GSPort
   ApplDomain Online  2.3.1.0   No  12347 12348
   ```

2. To display general information about the peer domains that **nodeA** belongs to, with the default delimiter and without the heading, run this command on **nodeA**:

   ```
   lsrpdomain -xd
   ```

   The output will look like this:

   ```
   ApplDomain:Online:2.3.1.0:No:12347:12348:
   ```

3. To display general information about the peer domains that**nodeA** belongs to in long format, run this command on **nodeA**:

   ```
   lsrpdomain -l
   ```

   The output will look like this:

   ```
   Name    = ApplDomain
   OpState    = Online
   RSCTActiveVersion = 2.3.1.0
   MixedVersions   = No
   TSPort    = 12347
   GSPort    = 12348
   ```

Location

/usr/sbin/rsct/bin/lsrpdomain Contains the **lsrpdomain** command

Related Information

Commands: **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmrpdomain**, **startrpdomain**, **stoprpdomain**

---

lsrpnode Command

Purpose

Displays information about one or more of the nodes that are defined in the online peer domain.

Syntax

lsrpnode [**–o** | **–O** | **–L**] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] [*node_name*]

lsrpnode [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–p** *domain_name*] [**–h**] [**–TV**]

Description

The **lsrpnode** command displays information about one or more of the nodes that are defined in the online peer domain. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify a node name, the command displays information about that node only. The **-o** and **-O** flags also limit the information this command displays. The **-o** flag displays information only about nodes that are online. The **-O** flag displays information only about nodes that are offline.

By default, the **lsrpnode** command displays information in table format (**-t**).

Some of the node information that is displayed follows:

Name Specifies the name of the node in the peer domain.

OpState Specifies the state of the node.

RSCTVersion Specifies the version of RSCT that is active in the node.

Flags

-o Displays information about the nodes that are online in the peer domain.

-O Displays information about the nodes that are offline in the peer domain.

-L Displays information about the local node only.

–p *domain_name*
Displays information about nodes defined in an *offline* peer domain that the local node belongs to. (By default, the **lsrpnode** command displays information about the nodes that are defined in the domain where you are currently *online*.) However, this information might not reflect changes that are made to the domain after the local node is taken offline, because an offline node might not have the latest configuration.

The **-p** flag ignores the **CT_CONTACT** environment variable.

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default format.

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x Excludes the header (suppresses header printing).

-h Writes the command's usage statement to standard output.

-T Writes the command's trace messages to standard error. For the IBM Support Center's use only.

-V Writes the command's verbose messages to standard output.

Parameters

node_name Specifies one or more nodes to display information about. You can specify a node name or a substring of a node name for this parameter. If you specify a substring, the command displays information about any defined node with a name that contains the substring.

Security

The user of the **lsrpnode** command needs read permission for the **IBM.PeerNode** resource class on the node this command runs on. 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 in the peer domain.

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

1. To display general information about the nodes in the online peer domain that **nodeA** belongs to, run this command on **nodeA**:

   ```
   lsrpnode
   ```

   The output will look like this:

   ```
   Name  OpState  RSCTVersion
   nodeA  online  2.3.1.0
   nodeB  online  2.3.1.0
   nodeC  offline  2.3.1.0
   ```

2. To display general information about the nodes in the online peer domain that **nodeA** belongs to, with the default delimiter and without the heading, run this command on **nodeA**:

   ```
   lsrpnode -xd
   ```

   The output will look like this:

   ```
   nodeA:online:2.3.1.0:
   nodeB:online:2.3.1.0:
   nodeC:offline:2.3.1.0:
   ```

3. To display general information about the nodes in the online peer domain that **nodeA** belongs to, run this command on **nodeA**:

   ```
   lsrpnode -l
   ```

   The output will look like this:

   ```
   Name        = nodeA
   OpState     = online
   RSCTVersion = 2.3.1.0

   Name        = nodeB
   OpState     = online
   RSCTVersion = 2.3.1.0

   Name        = nodeC
   OpState     = offline
   RSCTVersion = 2.3.1.0
   ```

Location

/usr/sbin/rsct/bin/lsrpnode Contains the **lsrpnode** command

Related Information

Commands: **addrpnode**, **preprpnode**, **startrpnode**, **stoprpnode**

---

lsrset Command

Purpose

Displays system rset contents.

Syntax

```
lsrset [ -f ] [ -v | -o ] [ -r rsetname | -n namespace | -a ]
```

or

```
lsrset [ -P ] [ -v | -o ] -p pid
```

Description

The **lsrset** command displays information contained in rsets stored in the system registry or rsets attached to a process.

Flags

-f	Displays rset owner, group, and mode data.
-v	Verbose mode. Displays resources contained in the rset, rset owner, group and mode data.
-o	Displays only the online resources contained in the rset. The default is to display all resources.
-p *pid*	Displays the effective rset attached to this process.
-r *rsetname*	Displays the rset with this name in the system registry. The name consists of a namespace and an *rsname* separated by a *"/"* (slash). Both the *namespace* and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.
-n *namespace*	Displays all rsets in this *namespace* in the system registry.
-a	Displays all rsets in the system registry.
-P	Displays the partition rset attached to the specified process.

Examples

1. To display all resources for all rsets in the system registry, type:
   ```
   lsrset -v -a
   ```
2. To display a summary of the effective rset attached to pid 28026, type:
   ```
   lsrset -p 28026
   ```
3. To display the online resources in the effective rset attached to pid 28026, type:
   ```
   lsrset -o -p 28026
   ```
4. To display all the resources in the effective rset attached to pid 28026, type:
   ```
   lsrset -v -p 28026
   ```
5. To display online resources for all rsets in the system registry, type:
   ```
   lsrset -a -o
   ```

Files

/etc/bin/lsrset	Contains the **lsrset** command

Related Information

The **attachrset**, **detachrset**, **execrset**, **mkrset**, and **rmrset** commands.

---

lsrsrc Command

Purpose

Lists (or displays) resources or a resource class.

Syntax

lsrsrc [**–s** *"selection_string"*] [**–A p** | **d** | **b**] [**–p** *property*] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–a**] [**–h**] [**–TV**] [*resource_class* [*attr*...]]

lsrsrc [**–s** *"selection_string"*] **–r** [**–l | –t | –d | –D** *delimiter*] [**–x**] [**–h**] [**–TV**] [*resource_class*]

lsrsrc –c [**–A p | d | b**] [**–p** *property*] [**–l | –t | –d | –D** *delimiter*] [**–x**] [**–a**] [**–h**] [**–TV**] *resource_class* [*attr*...]

Description

The **lsrsrc** command is used to list the persistent and dynamic attributes and their values of either a resource class or a resource.

When no *attr* parameter is specified, only attributes that are defined as public are displayed. Use the **-p** flag to override this default. When no *attr* parameter is specified, the **-A p** | **d** | **b** flag controls whether only persistent, or only dynamic, or both persistent and dynamic attributes and their values are displayed.

When one or more attribute names are specified, the attribute names specified and their values are displayed in the order specified, provided that each of the specified attribute names is valid.

To get a list of all of the resource classes, enter the **lsrsrc** command with no parameters.

Specify the **-c** flag to display a list of the resource class attributes and values.

Specify the **-r** flag to display only the resource handles associated with the resources for the specified resource class.

By default, the resource attributes and values are displayed in long format. Use the **-t**, **-d**, or **-D** flag for the resources to be displayed in tabular or delimiter formatted output.

For best performance, specify either the **-A p** flag or only persistent attributes as parameters.

Note: Any attribute that has a data type defined as ct_none (for example, a Quantum) is not listed by the **lsrsrc** command. RMC does not return attribute values for attributes that are defined as Quantum. To list attribute definitions, use the **lsrsrcdef** command.

Flags

–a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lsrsrc -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–A p | d | b
Specifies an attribute type. By default only persistent attributes are displayed. This flag can be used only when no attribute names are specified on the command line.

p Displays only persistent attributes.

d Displays only dynamic attributes.

b Displays both persistent and dynamic attributes.

For best performance, specify the **-A p** flag.

–c Displays the attributes for the resource class. By default, the resource attributes, not the resource class, are displayed. This flag overrides the **-r** flag.

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you want to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify

something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–l Specifies long formatted output. Each attribute is displayed on a separate line. This is the default display format. If the **lsrsrc** command is issued with the **-l** flag, but without a resource class name, the **-l** flag is ignored when the command returns the list of defined resource class names.

–p *property*
Displays attributes with the specified *property*. By default, only public attributes are displayed. To display all the attributes regardless of the property, use the **-p 0** flag. Use this flag in conjunction with the **-A** flag when no attributes are specified on the command line.

Persistent attribute properties:

0x0001	Read only
0x0002	Required for define
0x0004	Not valid for define
0x0008	Optional for define
0x0010	Selectable
0x0020	Public

Dynamic attribute properties:

0x0020	Public

A decimal or hexadecimal value can be specified for the property. To display attributes and their values for all attributes that have one or more properties, ″OR″ the properties of interest together and then specify the ″ORed″ value with the **-p** flag. For example, to display attributes and their values for all persistent attributes that are either required for define or optional for define, enter:

```
-p 0x0a
```

–r Displays the resource handles for the resources that match the specified selection string or all resources when no selection string is specified.

–s ″*selection_string*″
Specifies a selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

```
-s 'Name == "testing"'
-s 'Name ?= "test"'
```

Only persistent attributes may be listed in a selection string. For information on how to specify selection strings, see the *RSCT Administration Guide*.

–t Specifies tabular formatted output. Each attribute is displayed in a separate column, one resource per line.

–x Suppresses header printing.

–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

attr... Specifies the attribute name. Both persistent and dynamic attribute names can be specified to

control which attributes are displayed and their order. Zero or more attributes can be specified. Attributes must be separated by spaces. If no attribute names are specified, the **-A p | d | b** flag controls whether only persistent attributes, only dynamic attributes, or both persistent and dynamic attributes are displayed. When no attribute names are specified, only attributes that are defined as public are displayed. Use the **-p** flag to override this default.

resource_class
Specifies the name of the resource class with the resources that you want to display. Zero or one *resource_class* parameters can be specified. If no *resource_class* parameter is specified, a list of all resource class names is displayed.

Security

The user needs read permission for the *resource_class* specified in **lsrsrc** to run **lsrsrc**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0 Command has run successfully.

1 Error occurred with RMC.

2 Error occurred with CLI script.

3 Incorrect flag on command line.

4 Incorrect parameter on command line.

5 Error occurred with RMC that was based on faulty command line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. 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.

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

1. To list the names of all of the resource classes, enter:

   ```
   lsrsc
   ```

   The output will look like this:

   ```
   class_name
   "IBM.Association"
   "IBM.Condition"
   'IBM.EventResponse"
   "IBM.Host"
   "IBM.Ethernet"
   "IBM.TokenRing"
   ...
   ```

2. To list the persistent attributes for resource IBM.Host that have 4 processors, enter:

   ```
   lsrsc -s "NumProcessors == 4" -A p -p 0 IBM.Host
   ```

   The output will look like this:

   ```
   Resource Persistent Attributes for: IBM.Host
   resource 1:
           Name           = "c175n05.ppd.pok.ibm.com"
           ResourceHandle = "0x4008 0x0001 0x00000000 0x0069684c 0x0d7f55d5 0x0c32fde3"
           Variety        = 1
           NodeList       = {1}
           NumProcessors  = 4
           RealMemSize    = 1073696768
   ```

3. To list the public dynamic attributes for resource IBM.Host on node 1, enter:

   ```
   lsrsc -s 'Name == "c175n05.ppd.pok.ibm.com"' -A d IBM.Host
   ```

   The output will look like this:

   ```
   Resource Dynamic Attributes for: IBM.Host
   resource 1:
           ProcRunQueue        = 1.03347987093142
           ProcSwapQueue       = 1.00548852941929
           TotalPgSpSize       = 65536
           TotalPgSpFree       = 65131
           PctTotalPgSpUsed    = 0.61798095703125
           PctTotalPgSpFree    = 99.3820190429688
           PctTotalTimeIdle    = 0
           PctTotalTimeWait    = 51.5244382399734
           PctTotalTimeUser    = 12.8246006482343
           PctTotalTimeKernel  = 35.6509611117922
           PctRealMemFree      = 66
           PctRealMemPinned    = 4
           RealMemFramesFree   = 173361
           VMPgInRate          = 0
           VMPgOutRate         = 0
           VMPgFaultRate       = 0
            ...
   ```

4. To list the Name, Variety, and ProcessorType attributes for the IBM.Processor resource on all the online nodes, enter:

   ```
   lsrsc IBM.Processor Name Variety ProcessorType
   ```

   The output will look like this:

```
Resource Persistent Attributes for: IBM.Processor
resource 1:
        Name          = "proc3"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 2:
        Name          = "proc2"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 3:
        Name          = "proc1"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 4:
        Name          = "proc0"
        Variety       = 1
        ProcessorType = "PowerPC_604"
```

5. To list both the persistent and dynamic attributes for the resource class IBM.Condition, enter:

```
lsrsrc -c -A b -p 0 IBM.Condition
```

   The output will look like this:

```
Resource Class Persistent and Dynamic Attributes for: IBM.Condition
resource 1:
        ResourceType = 0
        Variety      = 0
```

Location

/usr/sbin/rsct/bin/lsrsrc Contains the **lsrsrc** command

Related Information

Commands: **lsrsrcdef**
Files: **rmccli**
Books: see the *RSCT Administration Guide* for information about RMC operations

---

lsrsrcdef Command

Purpose

Lists a resource or resource class definition.

Syntax

lsrsrcdef [**–h**] [**–c**] [**–A p | d**] [**–p** *property*] [**–e**] [**–s**] [**–l | –i | –t | –d | –D** *delimiter*] [**–x**] [**–TV**] [*resource_class* [*attr*...]]

Description

The **lsrsrcdef** command lists the definition of a resource class, or the persistent or dynamic attributes of a resource instance or a resource class.

Use **lsrsrcdef** with no parameters specified to get a list of all resource class names.

To see just the resource class definition, specify the **-c** flag and a *resource_class* parameter without specifying the **-A** flag.

To see the persistent or dynamic attribute definitions of the resource class, specify a *resource_class* parameter and the **-c** flag with the appropriate **-A** flag (**-A d** for dynamic attribute defintions or **-A p** for persistent attribute definitions). To see the persistent or dynamic attribute definitions for a resource, specify the appropriate **-A p | d** flag without the **-c** flag.

By default, when no *attr* parameters are specified on the command line, only the definitions for public attributes are displayed. To override this default, use the **-p** flag or specify the name of the attribute you want to display.

By default, for the attributes that contain descriptions, the descriptions are not displayed. Specify the **-e** flag to display the descriptions. Because some of the descriptions are very long, the default is not to display them.

Flags

–A p | d
Specifies the attribute type. Either persistent or dynamic attribute definitions may be displayed. Use this flag with the **-c** flag for the persistent or dynamic attribute definitions of the resource class.

p Displays only persistent attributes

d Displays only dynamic attributes

–c Displays the resource class definition. By default, the resource definition is displayed. Combine with the **-A p | d** flag to display the resource class definition for persistent or dynamic attributes. Use the **–c** flag alone, without the **-A p | d** flag, to see the resource class definition.

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–e Expands the descriptions. By default the definitions are displayed, but the textual descriptions are not displayed because they may be lengthy. Specify this flag to see both the definitions and descriptions.

–i Generates a template of the **Resource_Data_Input** file that can then, after appropriate editing, be used as input to the **mkrsrc** command. The output is displayed in long (stanza) format. All required and optional attributes that can be used to define a resource are displayed. The attribute data type is displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the **lsrsrcdef** command be directed to a file. This flag overrides the **–s** and **–A d** flags.

–l Specifies long formatted output, one entry per line. This is the default display format. If the **lsrsrcdef** command is issued with the **-l** flag, but without a resource class name, the **-l** flag is ignored when the command returns the list of defined resource class names.

–p *property*
Displays attribute definitions for attributes with the specified *property*. By default, only the definitions for public attributes are displayed. To display all attribute definitions regardless of the property, use the **-p 0** flag.

Persistent attribute properties:

0x0001 Read only

0x0002 Required for define

0x0004 Not valid for define

0x0008 Optional for define

0x0010 Selectable

0x0020 Public

Dynamic attribute properties:

0x0020 Public

A decimal or hexadecimal value can be specified for the property. To request the attribute definitions for all attributes that have one or more properties, ″OR″ the properties of interest together and then specify the ″ORed″ value with the **-p** flag. For example, to request the attribute definitions for all persistent attributes that are either required for define or optional for define, enter:

```
-p 0x0a
```

–s Displays the structured data definition. Specify this flag for the structured data definition to be expanded so that each element definition of the structured data attributes is displayed.

–t Specifies tabular formatted output. Each attribute is displayed in a separate column, one resource per line.

–x Suppresses header printing.

–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

attr If a *resource_class* parameter is specified, zero or more attribute names can be specified. If no *attr* parameter is specified, the definition for all the attributes for the resource are listed. Specific attribute names may be specified to control which attributes are displayed and their order. Specify only persistent attribute names when the **-A p** flag is used. Specify only dynamic attribute names when the **-A d** flag is used. Attributes must be separated by spaces.

resource_class
Specifies the name of the resource class whose attribute definitions are to be displayed. If no *resource_class* parameter is specified, a list of all of the resource class names is displayed.

Security

The user needs write permission for the *resource_class* specified in **lsrsrcdef** to run **lsrsrcdef**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0 Command has run successfully.

1 Error occurred with RMC.

2 Error occurred with CLI script.

3 Incorrect flag on command line.

4 Incorrect parameter on command line.

5 Error occurred with RMC that was based on faulty command line input.

Environment Variables

CT_CONTACT

When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. 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.

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

1. To list the names of all of the resource classes defined on the system, enter:

   ```
   lsrsrcdef
   ```

   The output will look like this:

   ```
   class_name
   "IBM.ATMDevice"
   "IBM.Association"
   "IBM.AuditLog"
   "IBM.AuditLogTemplate"
   "IBM.Condition"
   "IBM.EthernetDevice"
   "IBM.EventResponse"
   ...
   ```

2. To list the resource class definitions for resource IBM.Host, enter:

   ```
   lsrsrcdef -c IBM.Host
   ```

   The output will look like this:

   ```
   Resource Class Definition for: IBM.Host
   resource class 1:
           class_name         = "IBM.Host"
           class_id           = 8
           properties         = {"has_rsrc_insts","mtype_subdivided"}
           display_name       = ""
           description        = ""
           locator            = "NodeList"
           class_pattr_count  = 1
   ```

```
        class_dattr_count  = 3
        class_action_count = 0
        pattr_count        = 6
        dattr_count        = 47
        action_count       = 0
        error_count        = 0
        rsrc_mgr_count     = 1
rsrc_mgrs 1:
        mgr_name  = "IBM.HostRM"
        first_key = 1
        last_key  = 1
```

3. To list the resource class persistent attribute definitions for resource IBM.Host, enter:

```
lsrsrcdef -c -A p -p 0 IBM.Host
```

   The output will look like this:

```
Resource Class Persistent Attribute Definitions for: IBM.Host
attribute 1:
        program_name              = "Variety"
        display_name              = ""
        group_name                = ""
        properties                = {"read_only","invalid_for_define"}
        description               = ""
        attribute_id              = 0
        group_id                  = 255
        data_type                 = "uint32"
        variety_list              = {{1..1}}
        variety_count             = 1
        default_value             = 0
```

4. To list the resource persistent attribute definitions and descriptions for resource IBM.Host, enter:

```
lsrsrcdef -A p -p 0 -e IBM.Host
```

   The output will look like this:

```
Resource Persistent Attribute Definitions for: IBM.Host
attribute 1:
        program_name       = "Name"
        display_name       = "Name"
        group_name         = "General"
        properties         = {"required_for_define","public","selectable"}
        description        = "Identifies the current name of the host
                              as returned by command."
        attribute_id       = 0
        group_id           = 0
        data_type          = "char_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = ""
attribute 2:
        program_name       = "ResourceHandle"
        display_name       = "Resource Handle"
        group_name         = "Internal"
        properties         = {"read_only","invalid_for_define","selectable"}
        description        = "A globally unique handle that identifies the host.
                              Every resource is assigned a resource handle,
                              which is used internally for identifying and
                              locating each resource.  The resource handle
                              is fixed in size and avoids the problems of
                              name space collisions across different types
                              of resources."
        attribute_id       = 1
        group_id           = 255
        data_type          = "rsrc_handle_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
```

```
attribute 3:
        program_name       = "Variety"
        display_name       = "Variety"
        group_name         = "Internal"
 ...
```

5. To list the public dynamic attributes for resource IBM.Host, enter:

```
lsrsrcdef -A d IBM.Host
```

The output will look like this:

```
Resource Dynamic Attribute Definitions for: IBM.Host
attribute 1:
        program_name           = "ProcRunQueue"
        display_name           = ""
        group_name             = ""
        properties             = {"public"}
        description            = ""
        attribute_id           = 1
        group_id               = 1
        data_type              = "float64"
        variable_type          = 0
        variety_list           = {{1..1}}
        variety_count          = 1
        init_value             = 0
        min_value              = 0
        max_value              = 100
        expression             = "(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)"
        expression_description = ""
        rearm_expression       = "ProcRunQueue < 50"
        rearm_description      = ""
        PTX_name               = ""
attribute 2:
...
```

Location

/usr/sbin/rsct/bin/lsrsrcdef Contains the **lsrsrcdef** command

Related Information

Commands: **lsrsrc**, **mkrsrc**

Files: **Resource_Data_Input**, **rmccli**

Books: see the *RSCT Administration Guide* for information about RMC operations

---

lssavevg Command

Purpose

Lists or restores the contents of a volume group backup on a specified media.

Syntax

lssavevg [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-l**] [**-r**] [**-s**] [**-d** *path*] [**-B**] [**-D**] [**-L**] [**-V**] [*file_list*]

Description

The **lssavevg** command lists the contents of a volume group backup from tape, file, CD-ROM, or other source and can be used to restore files from a valid backup source. The **lssavevg** command also works for multi-volume backups such as multiple CDs, DVDs, or tapes.

The **lssavevg -r** and **restorevgfiles** commands perform identical operations and should be considered interchangeable.

Flags

-a Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You may need to alter the block size if necessary to read the backup. The **-a** flag is valid only when a tape backup is used.

-b *blocks* Specifies the number of 512-byte blocks to read in a single input operation, as defined by the *blocks* parameter. If the *blocks* parameter is not specified, the number of blocks read will default to 100.

-B Prints the volume group backup log to **stdout**.

This flag will display the past 256 backups (roughly). The log is in alog format and is kept in **/var/adm/ras/vgbackuplog**. Each line of the log is a semicolon-separated list of the file or device name, the command used to make backup, date, shrink size, full size of the backup, and recommended maintenance level (if any).
Note: The shrink size is the size of the data on all filesystems. The full size is total size of each filesystem (unused + data).

-c Produces colon-separated output. This flag only works with the **-l** and **-L** flags.

-d *path* Specifies the directory path to which the files will be restored, as defined by the *path* parameter. If the **-d** parameter is not used, the current working directory is used. This can be a problem if the current working directory is root. We recommend writing to a temporary folder instead of to root.

-D Produces debug output.

-l Displays useful information about a volume group backup.

This flag requires the **-f** *device* flag. This flag causes **lssavevg** to display information such as volume group, date and time backup was made, uname output from backed up system, oslevel, recommended maintenance level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all filesystems. The full size is the total size of each filesystem (unused + data). The **-l** flag also displays the logical volume and filesystem information of the backed up volume group, equivalent to running ″**lsvg -l** *vgname*″.

-L Displays lpp fileset information about a **mksysb** backup only.

This flag requires the **-f** *device* flag and displays the equivalent information to that produced by invoking ″**lslpp -l**″ on the running backed up system. This flag does not produce output about any volume group backup other than that produced by **mksysb**.

-f *device* Specifies the type of device containing the backup (file, tape, CD-ROM, or other source) as defined by the *device* parameter. When **-f** is not specified, *device* will default to **/dev/rmt0**.

-r Specifies to restore the backup files, as defined by the *file-list* parameter. If the *file-list* parameter is not specified, then all files in the backup will be restored. If the **-r** flag is not used, then executing the **lssavevg** command only lists the files in the specified backup.

-s Specifies that the backup source is a user volume group and not rootvg.

-V Verifies a tape backup.

This flag requires the **-f** *device* flag and works for tape devices only. The **-V** flag causes **lssavevg** to verify the readability of the header of each file on the volume group backup and print any errors that occur to **stderr**.

Parameters

file_list Identifies the list of files to be restored. This parameter is used only when the **-r** flag is specified. The full path of the files relative to the current directory should be specified in the space-separated list. All files in the specified directory will be restored unless otherwise directed. If you are restoring all files in a directory, we recommend writing to a temporary folder instead of to root.

Examples

1. To list the contents of the system backup located on the default device **/dev/rmt0**, enter:
   ```
   lssavevg
   ```
2. To list the contents of the system backup located on device **/dev/cd1**, enter:
   ```
   lssavevg -f /dev/cd1
   ```
3. To list the contents of the system backup located on device **/dev/cd1**, which is a user volume group that is not rootvg, enter:
   ```
   lssavevg -f /dev/cd1 -s
   ```
4. To restore **/etc/filesystems** from the system backup located on device **/dev/cd1**, enter:
   ```
   lssavevg -f /dev/cd1 -r ./etc/filesystems
   ```
5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is located on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter:
   ```
   lssavevg -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```
6. To display colon separated lpp information about a **mksysb** backup tape located on **/dev/rmt0**, enter the following:
   ```
   lsmksysb -Lc -f /dev/rmt0
   ```
7. To display the volume group backup log to **stdout**, enter:
   ```
   lssavevg -B
   ```
8. To list volume group and general backup data about a backup located at **/tmp/mybackup**, enter:
   ```
   lssavevg -l -f /tmp/mybackup
   ```
9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter:
   ```
   lsmksysb -V -f /dev/rmt0
   ```

Files

/usr/bin/lssavevg	Contains the **lssavevg** command

Related Information

The **restorevgfiles** command.

---

lssec Command

Purpose

Lists attributes in the security stanza files.

Syntax

lssec [**-c**] [**-f** *File*] [**-s** *Stanza*] [**-a** *Attribute* ...]

Description

The **lssec** command lists attributes stored in the security configuration stanza files. The following security configuration files contain attributes that you can specify with the *Attribute* parameter:

- **/etc/security/environ**
- **/etc/security/group**
- **/etc/security/lastlog**
- **/etc/security/limits**
- **/etc/security/login.cfg**
- **/usr/lib/security/mkuser.default**

- **/etc/security/passwd**
- **/etc/security/portlog**
- **/etc/security/user**

When listing attributes in the **/etc/security/environ**, **/etc/security/lastlog**, **/etc/security/limits**, **/etc/security/passwd**, and **/etc/security/user** files, the stanza name specified by the *Stanza* parameter must be either a valid user name or `default`. When listing attributes in the **/etc/security/group** file, the stanza name specified by the *Stanza* parameter must be either a valid group name or `default`. When listing attributes in the **/usr/lib/security/mkuser.default** file, the *Stanza* parameter must be either `admin` or `user`. When listing attributes in the **/etc/security/portlog** file, the *Stanza* parameter must be a valid port name. When listing attributes in the **/etc/security/login.cfg** file, the *Stanza* parameter must be either a valid port name, a method name, or the **usw** attribute.

You cannot list the **password** attribute of the **/etc/security/passwd** file with the **lssec** command.

Only the root user or a user with PasswdAdmin authorization can list the lastupdate and flags attributes for administrative users.

Flags

-c	Specifies that the output should be in colon-separated format.
-f *File*	Specifies the name of the stanza file to list.
-s *Stanza*	Specifies the name of the stanza to list.
-a *Attribute*	Specifies the attribute to list.

Security

Access Control: This command grants execute access only to the root user and the security group. The command has the trusted computing base attribute and runs the **setuid** subroutine for the root user to access the security databases.

Files Accessed:

Mode	File
r	/etc/security/environ
r	/etc/security/group
r	/etc/security/lastlog
r	/etc/security/limits
r	/etc/security/login.cfg
r	/usr/lib/security/mkuser.default
r	/etc/security/passwd
r	/etc/security/portlog
r	/etc/security/user

Examples

1. To list the number of unsuccessful login attempts by the root user since the last successful login of the root user, enter:

   ```
   lssec -f /etc/security/lastlog -s root -a unsuccessful_login_count
   ```

   The system displays the result as follows:

   ```
   root unsuccessful_login_count=15
   ```

2. To list the times that logins are allowed on the **/dev/tty2** port, enter:

   ```
   lssec -f /etc/security/login.cfg -s /dev/tty2 -a logintimes
   ```

The system displays the result as follows:

```
/dev/tty0 logintimes=!january1,!july4,!december25
```

3. To list the default setting for the **tpath** attribute and the **ttys** attribute in colon format,
4. enter:

```
lssec -c -f /etc/security/user -s default -a tpath -a ttys
```

The system displays the result as follows:

```
#name:tpath:ttys
default:nosak:ALL
```

Files

/usr/bin/lssec	Specifies the path to the **lssec** command.
/etc/security/environ	Contains the environment attributes of users.
/etc/security/group	Contains extended attributes of groups.
/etc/security/lastlog	Defines the last login attributes for users.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/login.cfg	Contains port configuration information.
/usr/lib/security/mkuser.default	Contains the defaults values for new users.
	Contains password information.
/etc/security/portlog	Contains unsuccessful login attempt information for each port.
/etc/security/user	Contains the extended attributes of users.

Related Information

The **chgroup** command, **chsec** command, **chuser** command, **grpck** command, **login** command, **lsgroup** command, **lsuser** command, **mkgroup** command, **mkuser** command, **passwd** command, **pwdck** command, **rmgroup** command, **rmuser** command, **su** command, **usrck** command.

The **getgroupattr** subroutine, **getportattr** subroutine, **getuserattr** subroutine, **getuserpw** subroutine, **putgroupattr** subroutine, **putportattr** subroutine, **putuserattr** subroutine, **putuserpw** subroutine.

List of Security and Auditing Subroutines in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

---

lssensor Command

Purpose

Displays the event sensor commands that have been added to the resource monitoring and control (RMC) subsystem.

Syntax

lssensor [**–v** | **–V**] [**–a** | **–n** *host1*[**,***host2*...]] [**–A** | **–h** | *name1* [*name2*...]]

Description

The **lssensor** command displays the attributes of the sensor identified by *name*. If *name* is omitted, the **lssensor** command lists the names of all the sensors defined to RMC. This command returns the error messages and exit status of the underlying **lsrsrc** RMC command.

The **lssensor** command can be run on the management server or on any node. It operates on the local node where it is run unless the CT_CONTACT environment variable is set. If you want **lssensor** to run on all of the nodes in a domain, use the CT_MANAGEMENT_SCOPE environment variable. See the **Environment variables** section for more information.

See the **mksensor** command for a more complete explanation of RMC sensors.

Flags

–v | –V
Writes the command's verbose messages to standard output. Use this flag to see the underlying RMC command that is being run. This can help to interpret error messages that are displayed.

–a Lists sensors from all nodes in the domain.

–n *host1***[,***host2***...]**
Specifies which nodes the sensor should be listed for. By default, sensors on the local node are listed. This flag is only appropriate in a management domain or a peer domain.

–A Displays all attributes of all sensors.

–h Writes the command's usage statement to standard output.

Parameters

name1 **[***name2***...]**
Specifies one or more names of sensors for which to display all attributes.

Security

The user needs read permission for the **IBM.Sensor** resource class in order to run **lssensor**. 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 has run successfully.

1 An incorrect combination of flags and parameters has been entered.

If an error occurs in the RMC layer, the RMC return code will be returned as the exit status.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE
Determines the management scope used for the command's RMC daemon session.

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.

Examples

1. To list the names of all of the sensors, enter:
   ```
   lssensor
   ```
2. To list the names and attributes of all sensors, enter:
   ```
   lssensor -A
   ```
3. To list the attributes of the sensor called **NumLogins**, enter:
   ```
   lssensor NumLogins
   ```

Location

/usr/sbin/rsct/bin/lssensor Contains the **lssensor** command

Related Information

Commands: **chsensor**, **lsrsrc**, **mksensor**, **rmsensor**

Books: see the *RSCT Administration Guide* for information about the ACL authorization file

---

lsslot Command

Purpose

Displays dynamically reconfigurable slots, such as hot plug slots, and their characteristics.

Syntax

lsslot -c *ConnectorType* [**-a** | **-o** | **-l** *DeviceName* | **-s** *Slot*] [**-F** *Delimiter*]

Description

The **lsslot** command displays all the specified hot plug slots and their characteristics. Hot plug slots are the plug-in points for connecting entities that can be added and removed from the system without turning the system power off or rebooting the operating system. The **-c** flag is required. It specifies the type of hot plug connector, for example, pci for hot pluggable PCI adapters. You can display only the empty, that is, available, hot plug slots with the **-a** flag, the occupied slots with the **-o** flag, or a specific slot by using the **-s** flag. The **-l** flag can be used to locate the slot associated with specified *DeviceName*, as listed by the **lsdev** command.

The output of the **lsslot** command is dependent on the *ConnectorType* and the platform on which the command is executed. The characteristics of a slot may include the following:

- Slot name or identification
- Connector type or slot description, for example, a PCI hot plug slot
- Connected device name(s), for example, scsi0, ent0

Flags

-a Displays available hot plug slots and their characteristics. Available slots are those slots that do not have a hot plug device connected.

-c *ConnectorType* Displays the slots of the specified *ConnectorType*. *ConnectorType* identifies the type of hot plug connector. For example, the *ConnectorType* for a hot plug PCI slot is pci. This flag is required.

-F *Delimiter*	Specifies a single character to delimit the output. The heading is not displayed and the columns are delimited by the *Delimiter* character.
-l *DeviceName*	Displays the characteristics of the slot to which *DeviceName* is connected. The *DeviceName* is the logical device name of the device connected to the slot, as listed by the lsdev command.
-o	Displays the characteristics of the occupied slots. Occupied slots have a hot plug device connected.
-s *Slot*	Displays characteristics for the specified *Slot.* The format of *Slot* is platform/connector_type dependent.

Examples

1. To list the available PCI hot plug slots, enter:

   ```
   lsslot -c pci -a
   ```

   The system displays a message similar to the following:

Slot name	Description	Device(s) Connected
U0.4-P1-I1	PCI 64 bit, 66MHz, 3.3 volt slot	empty
U0.4-P1-I2	PCI 64 bit, 66MHz, 3.3 volt slot	empty
U0.4-P1-I3	PCI 64 bit, 66MHz, 3.3 volt slot	empty

2. To list the PCI hot plug slot associated with a scsi adapter named scsi1, enter:

   ```
   lsslot -c pci -l scsi1
   ```

   The system displays a message similar to the following:

Slot name	Description	Device(s) Connected
U0.4-P1-I1	PCI 64 bit, 33MHz, 5 volt slot	scsi1

3. To list all the PCI hot plug slots, enter:

   ```
   lsslot -c pci
   ```

   The system displays a message similar to the following:

Slot name	Description	Device(s) Connected
U0.4-P1-I1	PCI 64 bit, 33MHz, 3.3 volt slot	empty
U0.4-P1-I2	PCI 64 bit, 33MHz, 3.3 volt slot	scsi0
U0.4-P1-I3	PCI 64 bit, 33MHz, 3.3 volt slot	unknown
U0.4-P1-I5	PCI 64 bit, 33MHz, 3.3 volt slot	empty

   Slots that have *unknown* in the Device(s) Connected column have a device connected to the slot, but the device isn't in the ODM customized device (CuDv) database. This can be due to the device having been newly added but not configured yet, deleted with the **rmdev -d** command, or the system may not be installed with the software packages associated with the device.

Files

/usr/sbin/lsslot

Related Information

The drslot command, the lsdev command.

For information about Hot Plug Management and PCI Hot Plug Support for PCI Adapters, see PCI Hot Plug Management in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information about using PCI Hot Plug Support for PCI Adapters, see Managing Hot Plug Connectors in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

---

lssrc Command

Purpose

Gets the status of a subsystem, a group of subsystems, or a subserver.

Syntax

To Get All Status

lssrc [**-h** *Host*] **-a**

To Get Group Status

lssrc [**-h** *Host*] **-g** *GroupName*

To Get Subsystem Status

lssrc [**-h** *Host*] [**-l**] **-s** *Subsystem*

To Get Status by PID

lssrc [**-h** *Host*] [**-l**] **-p** *SubsystemPID*

To Get Subserver Status

lssrc [**-h** *Host*] [**-l**] **-t** *Type* [**-p** *SubsystemPID*] [**-o** *Object*] [**-P** *SubserverPID*]

To Get Subsystem Status in SMIT Format

lssrc -S [**-s** *Subsystem* | **-d**]

To Get Subserver Status in SMIT Format

lssrc -T [**-t** *Type*]

To Get Notify in SMIT Format

lssrc -N [**-n** *NotifyName*]

Description

The **lssrc** command sends a request to the System Resource Controller to get status on a subsystem, a group of subsystems, or all subsystems. The **lssrc** command sends a subsystem request packet to the daemon to be forwarded to the subsystem for a subserver status or a long subsystem status.

You can choose whether to request a short or long status for a subserver. When the **-l** flag is absent, the status request is assumed to be a short status. A short status of a subsystem, group of subsystems, or all subsystems is handled by the System Resource Controller.

When the **-l** flag is present for a subsystem, a status request is taken to the subsystem and the subsystem sends the status back. The **-l** flag is supported only for those subsystems not using signals as their communication method. For either a long or short status of a subserver, the subsystem is sent a status request packet, and the subsystem sends the status back.

Flags

-a	Lists the current status of all defined subsystem.
-d	Specifies that the default record is printed.

-g *GroupName*	Specifies a group of subsystems to get status for. The command is unsuccessful if the *GroupName* variable is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this status 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	Requests that a subsystem send its current status in long form. Long status requires that a status request be sent to the subsystem; it is the responsibility of the subsystem to return the status.
-n *NotifyName*	Specifies the name of a notify method.
-N	Specifies that the Object Data Manager (ODM) records are output in SMIT format for the notify object class.
-o *Object*	Specifies that a subserver *Object* variable is passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the *SubsystemPID* variable to get status for, or a particular instance of the subsystem to which the status subserver request is to be taken.
-P *SubserverPID*	Specifies that a *SubserverPID* variable is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to get status for. The *Subsystem* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* variable is not contained in the subsystem object class.
-S	Specifies that the ODM records are output in SMIT format for the subsystem object class.
-t *Type*	Requests that a subsystem send the current status of a subserver. The command is unsuccessful if the subserver *Type* variable is not contained in the subserver object class.
-T	Specifies that the ODM records are output in SMIT format for the subserver object class.

Examples

1. To get the status of all subsystems on the local machine, enter:

   ```
   lssrc -a
   ```

   This gets the status of all subsystems known on the local machine.
2. To get the status of all subsystems on a foreign host, enter:

   ```
   lssrc -h zork -a
   ```

   This gets the status of all subsystems known on the `zork` machine.
3. To get the status of the srctest subsystem, enter:

   ```
   lssrc -s srctest
   ```

   This gets the status of all instances of the `srctest` subsystem on the local machine.
4. To get the status of the subsystem by PID, enter:

   ```
   lssrc -p 1234
   ```

   This gets the status of the subsystem with the subsystem PID of `1234` on the local machine.
5. To get the status of the `tcpip` subsystem group, enter:

   ```
   lssrc -g tcpip
   ```

This gets the status of all instances of subsystems in the `tcpip` group on the local machine.

6. To get the status of the `tester` subserver, enter:

```
lssrc  -t tester  -p 1234
```

This gets the status of `tester` subserver that belongs to the `srctest` subsystem with the subsystem PID of `1234` on the local machine.

7. To get the status of the subsystem by PID, enter:

```
lssrc  -l  -p 1234
```

This gets the long status of the subsystem with the PID of `1234`.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify 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 **mkssys** command, **rmssys** command.

System Resource Controller Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* gives an explanation of subsystems, subservers, and the System Resource Controller.

---

lstun Command

Purpose

Lists tunnel definition(s).

Syntax

lstun [**-v 4**|**6**] [**-t** *tid_list*] [**-p manual**] [**-a**]

Description

Use the **lstun** command to list the tunnel definition(s) and their current status. This command can either list the tunnels in the tunnel database or in the active system.

Flags

-v	This flag specifies the IP version. For listing IP version 4 tunnel only, use the value of **4**. For listing IP version 6 tunnel only, use the value of **6**. If this flag is not used, both the version 4 and version 6 tunnels will be listed.
-t	Only list the tunnel definition and its current status for the tunnel whose tunnel ID is in **tid_list**. If this flag is not used, all the tunnel definitions and their current status will be listed.

-p Selects the type of the tunnel to be listed. Using the **-p** flag with the value of **manual** lists **manual** tunnels only. The **-p** flag is for listing tunnel definitions in the tunnel database only and thus is mutually exclusive with the **-a** flag.

-a Lists the tunnels active in the IP Security subsystem.

Related Information

The **gentun** command, **chtun** command, **imptun** command, **exptun** command, **mktun** command, and **rmtun** command.

---

lsuser Command

Purpose

Displays user account attributes.

Syntax

lsuser [**-R** *load_module*] [**-c** | **-f**] [**-a** *List*] { **ALL** | *Name* [,*Name*] *... }*

Description

The **lsuser** command displays the user account attributes. You can use this command to list all attributes of all the system users or all the attributes of specific users. Since there is no default parameter, you must enter the **ALL** keyword to see the attributes of all the users. By default, the **lsuser** command displays all user attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsuser** command lists as much information as possible.

> **Note:** If you have a Network Information Service (NIS) database installed on your system, some user information may not appear when you use the **lsuser** command.

By default, the **lsuser** command lists each user's attributes on one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the user attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-c** flag.

You can use the Users application in Web-based System Manager (wsm) to change user characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsusers** fast path to run this command.

Flags

-a *List* Lists the attributes to display. The *List* variable can include any attribute defined in the **chuser** command and requires a blank space between attributes. If you specify an empty list, only the user names are displayed.

-c Displays the user attributes in colon-separated records, as follows:

```
# name:  attribute1:  attribute2:  ...
  User:  value1:      value2:      ...
```

-f Displays the output in stanzas, with each stanza identified by a user name. Each *Attribute*=*Value* pair is listed on a separate line:

```
user:
       attribute1=value
       attribute2=value
       attribute3=value
```

-R *load_module* Specifies the loadable I&A module used to display the user account attributes.

Security

Access Control: This command should be a general user program with execute (x) access for all users. Since the attributes are read with the access rights of the user who invokes the command, some users may not be able to access all the information. This command should have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	/etc/passwd
r	/etc/security/user
r	/etc/security/user.roles
r	/etc/security/limits
r	/etc/security/environ
r	/etc/group
r	/etc/security/audit/config

Examples

1. To display the user `id` and group-related information about the `smith` account in stanza form, enter:

   ```
   lsuser -f -a id pgrp groups admgroups smith
   ```

   Information similar to the following appears:

   ```
   smith:
      ID=2457
      pgrp=system
      groups=system,finance,staff,accounting
      admgroups=finance,accounting
   ```

2. To display the user `id`, `groups`, and home directory of `smith` in colon format, enter:

   ```
   lsuser -c -a id home groups smith
   ```

   Information like the following appears:

   ```
   # name: ID:home:groups
   smith:  2457:/home/smith:system,finance,staff,accounting
   ```

3. To display all the attributes of user smith in the default format, enter:

   ```
   lsuser smith
   ```

   All the attribute information appears, with each attribute separated by a blank space.

4. To display all the attributes of all the users, enter:

   ```
   lsuser ALL
   ```

   All the attribute information appears, with each attribute separated by a blank space.

Files

/usr/sbin/lsuser	Contains the **lsuser** command.
/etc/passwd	Contains basic user information.
/etc/security/limits	Defines resource quotas and limits for each user.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains the administrative role attributes of users.
/etc/security/environ	Contains the environment attributes of users.
/etc/group	Contains basic group attributes.

/etc/security/audit/config Contains the audit configuration files.

Related Information

The **chfn** command, **chgroup** command, **chgrpmem** command, **chsh** command, **chuser** command, **lsgroup** command, **mkgroup** command, **mkuser** command, **passwd** command, **pwdadm** command, **rmgroup** command, **rmuser** command, **setgroups** command, **setsenv** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

For more information about administrative roles, refer to Administrative Roles Overview in *AIX 5L Version 5.2 Security Guide*.

---

lsvfs Command

Purpose

Lists entries in the **/etc/vfs** file.

Syntax

lsvfs { **-a** | *VfsName* }

Description

The **lsvfs** command lists entries in the **/etc/vfs** file. You can display information about a specific Virtual File System (VFS) type or all known VFS types.

Flag

-a Lists all stanzas in the **/etc/vfs** file, including the default stanza.

Parameter

VfsName Specifies the name of a virtual file system.

Examples

1. To list the vfs entry named `newvfs`, enter:
   ```
   lsvfs newvfs
   ```
2. To list all vfs types, enter:
   ```
   lsvfs -a
   ```

Files

/etc/vfs Contains descriptions of virtual file system types.

Related Information

The **chvfs** command, **crvfs** command, **rmvfs** command, **mount** command.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains file system types, management, structure, and maintenance.

Mounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains mounting files and directories, mount points, and automatic mounts.

---

lsvg Command

Purpose

Displays information about volume groups.

Syntax

lsvg [**-L**] [**-o**] | [**-n** *DescriptorPhysicalVolume*] | [**-i**] [**-l** | **-M** | **-p**] *VolumeGroup* ...

Description

The **lsvg** command displays information about volume groups. If you use the *VolumeGroup* parameter, only the information for that volume group is displayed. If you do not use the *VolumeGroup* parameter, a list of the names of all defined volume groups is displayed.

When information from the Device Configuration database is unavailable, some of the fields will contain a question mark (?) in place of the missing data. The **lsvg** command attempts to obtain as much information as possible from the description area when the command is given a logical volume identifier.

> **Note:** To determine a volume group's major number, use the **ls -al /dev/***VGName* command. This command lists the special device file that represents the volume group. The volume group major number is the same as the major device number of the special device file. For example, for a volume group named `ha1vg`, enter the following command:
>
> ```
> ls -al /dev/ha1vg
> ```
>
> This command returns the following:
>
> ```
> crw-rw---- 1 root system 52, 0 Aug 27 19:57 /dev/ha1vg
> ```
>
> In this example, the volume group major number is `52`.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsvg** fast path to run this command.

Flags

-L Specifies no waiting to obtain a lock on the Volume group.

> **Note** : If the volume group is being changed, using the **-L** flag gives unreliable date.

-p Lists the following information for each physical volume within the group specified by the *VolumeGroup* parameter:

Physical volume
A physical volume within the group.

PVstate
State of the physical volume.

Total PPs
Total number of physical partitions on the physical volume.

Free PPs
Number of free physical partitions on the physical volume.

Distribution
The number of physical partitions allocated within each section of the physical volume: outer edge, outer middle, center, inner middle, and inner edge of the physical volume.

-l Lists the following information for each logical volume within the group specified by the *VolumeGroup* parameter:

LV A logical volume within the volume group.

Type Logical volume type.

LPs Number of logical partitions in the logical volume.

PPs Number of physical partitions used by the logical volume.

PVs Number of physical volumes used by the logical volume.

Logical volume state
State of the logical volume. `Opened/stale` indicates the logical volume is open but contains partitions that are not current. `Opened/syncd` indicates the logical volume is open and synchronized. `Closed` indicates the logical volume has not been opened.

Mount point
File system mount point for the logical volume, if applicable.

-i Reads volume group names from standard input.

-M Lists the following fields for each logical volume on the physical volume:

`PVname:PPnum [LVname: LPnum [:Copynum] [PPstate]]`

PVname Name of the physical volume as specified by the system.

PPnum Physical partition number. Physical partition numbers can range from 1 to 1016.

LVname Name of the logical volume to which the physical partitions are allocated. Logical volume names must be system-wide unique names, and can range from 1 to 64 characters.

LPnum Logical partition number. Logical partition numbers can range from 1 to 64,000.

Copynum
Mirror number.

PPstate
Only the physical partitions on the physical volume that are not current are shown as stale.

-n *DescriptorPhysicalVolume* Accesses information from the descriptor area specified by the *DescriptorPhysicalVolume* variable. The information may not be current, since the information accessed with the -n flag has not been validated for the logical volumes. If you do not use the -n flag, the descriptor area from the physical volume that holds the most validated information is accessed, and therefore the information displayed is current. The volume group need not be active when you use this flag.

-o Lists only the active volume groups (those that are varied on). An active volume group is one that is available for use.

Information displayed if you do not specify any flags:

`Volume group`	Name of the volume group. Volume group names must be unique systemwide and can range from 1 to 15 characters.
`Volume group state`	State of the volume group. If the volume group is activated with the **varyonvg** command, the state is either `active/complete` (indicating all physical volumes are active) or `active/partial` (indicating some physical volumes are not active). If the volume group is not activated with the **varyonvg** command, the state is `inactive`.
`Permission`	Access permission: `read-only` or `read-write`.
`Max LVs`	Maximum number of logical volumes allowed in the volume group.
`LVs`	Number of logical volumes currently in the volume group.
`Open LVs`	Number of logical volumes within the volume group that are currently open.
`Total PVs`	Total number of physical volumes within the volume group.
`Active PVs`	Number of physical volumes that are currently active.
`VG identifier`	The volume group identifier.
`PP size`	Size of each physical partition.
`Total PPs`	Total number of physical partitions within the volume group.
`Free PPs`	Number of physical partitions not allocated.
`Alloc PPs`	Number of physical partitions currently allocated to logical volumes.
`Quorum`	Number of physical volumes needed for a majority.
`VGDS`	Number of volume group descriptor areas within the volume group.
`Auto-on`	Automatic activation at IPL (`yes` or `no`).

Concurrent	States whether the volume group is Concurrent Capable or Non-Concurrent Capable.
Auto-Concurrent	States whether you should autovary the Concurrent Capable volume group in concurrent or non-concurrent mode. For volume groups that are Non-Concurrent Capable, this value defaults to Disabled.
VG Mode	The vary on mode of the volume group: Concurrent or Non-Concurrent.
Node ID	Node id of this node if volume group is varied on in concurrent node.
Active Nodes	Node ids of other concurrent nodes that have this volume group varied on.
Max PPs Per PV	Maximum number of physical partitions per physical volume allowed for this volume group.
Max PVs	Maximum number of physical volumes allowed in this volume group.
LTG size	Logical track group size, in number of kilobytes, of the volume group.
BB POLICY	Bad block relocation policy of the volume group.
SNAPSHOT VG	Snapshot volume group name if the snapshot volume group is active else snapshot volume group identifier.
PRIMARY VG	Original volume group name of a snapshot volume group if the original volume group is active else original volume group identifier.

Examples

1. To display the names of all active volume groups, enter:

   ```
   lsvg  -o
   ```

2. To display the names of all volume groups within the system, enter:

   ```
   lsvg
   ```

3. To display information about volume group `vg02`, enter:

   ```
   lsvg vg02
   ```

   The characteristics and status of both the logical and physical partitions of volume group `vg02` are displayed.

4. To display the names, characteristics, and status of all the logical volumes in volume group `vg02`, enter:

   ```
   lsvg  -l vg02
   ```

Files

/usr/sbin	Contains the directory where the **lsvg** command resides.

Related Information

The **chvg** command, **lspv** command, **lslv** command, **varyonvg** command.

Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the Logical Volume Manager, physical volumes, logical volumes, volume groups, organization, ensuring data integrity, and allocation characteristics.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices* explains the structure, main menus, and tasks that are done with SMIT.

lsvirprt Command

Purpose

Displays the attribute values of a virtual printer.

Syntax

lsvirprt [**-q** *QueueName* **-d** *DeviceName* { [**-f** *Format*] [**-n**] [**-a** *AttributeName* | **-s** *SectionName*] ... | **-i** | **-D** }]

Description

The **lsvirprt** command displays the attribute values for the virtual printer assigned to the *PrintQueueName* and *QueueDeviceName* variables.

The **lsvirprt** command becomes interactive if no flags are specified with the command. A list of print queue names is displayed, and a prompt appears requesting that the desired print queue name be selected. After a valid print queue name is selected, a prompt appears requesting that attribute names be entered. If an attribute name of * (asterisk) is entered, a list of all attributes is displayed.

> **Note:** Attribute names for default values of the **qprt** command line flags can be specified by entering the flag letters. For example, to view the default value for the **-w** flag (page width), enter the **w** attribute name. All other attribute names must be 2 characters long.

You can use the Printer Queues application in Web-based System Manager (wsm) to change printer characteristics. You could also use the System Management Interface Tool (SMIT) **smit lsvirprt** fast path to run this command.

Flags

-a *AttributeName*	Specifies the name of an attribute for which information is to be displayed. The flag cannot be used with the **-s** flag. The **-a** flag can be specified many times to list multiple attributes. The *AttributeName* value can be a single-character name (for example, **j**), a simple two-character name (for example, **ci**), or a regular expression that specifies multiple attributes (for example, **^i.***.)
-d *QueueDeviceName*	Specifies the name of the queue device to which the virtual printer is assigned. This flag is optional, but can be specified only if the **-q** flag is specified.
-D	Displays data streams supported by a given queue and queue device name variable values. The **-D** flag displays the default data stream first and all other supported data streams in alphabetical order.

-f *Format*
: Specifies the display format for attribute information. Attribute information includes the attribute value, limits field, and attribute description. The *Format* value is specified in **printf** format. The **-f** *Format* option also supports the following predefined set of position arguments:

  %1$[*.*]s
  : Message catalog name

  %2$[*.*]d
  : Message number

  %3$[*.*]s
  : Attribute name

  %4$[*.*]s
  : Limits field

  %5$[*.*]s
  : Attribute value

  %6$[*.*]s
  : Attribute description

  %7$c Second character of attribute name.

-i
: Sets the command to interactive mode. The **-q** and **-d** flags must be specified with the **-i** flag. If values have been assigned to the *QueueName* and *DeviceName* variables, the command does not prompt for the queue and device names and accepts attribute names interactively.

-n
: Displays only the specified attributes that have nonnull values.

-s *SectionName*
: Specifies a section name in the virtual printer attribute database of the specified queue and queue device. The *SectionName* values begin with two underscores and contain three characters that identify a section. For example, the name of a section that contains all the flag attributes is __**FLG**. The **-s** flag can not be used with the **-a** flag. This option can be repeated to list multiple attributes. The *SectionName* variable value can be a regular expression.

-q *PrintQueueName*
: Specifies the name of the print queue to which the virtual printer is assigned. This flag is optional, but can be specified only if the **-d** flag is specified.

Examples

1. To show the attribute values for the `w` (default page width) and `si` (user to receive the ″Intervention Required″ messages) attributes for the virtual printer assigned to the `mypro` queue device on the `proq` print queue, enter:

   ```
   lsvirprt -dmypro -qproq -a w -a si
   ```

   The output from this command is:

   ```
   Name            Description                     Value
   _w              COLUMNS per page                136
   si              USERS to get intervention messages
   ```

2. To show the same attributes in Example 1, but the be prompted for the flag values, enter:

   ```
   lsvirpt
   ```

   The output from this command is:

   ```
   1       e4039c          @piobe          ibm4039 (PCL Emulation)
   2       e4039s          @piobe          ibm4039 (PostScript)
   3       fjzhp4s         jzfile          hplj-4 (PostScript)
   4       hpc14           hp@pc15         hplj-4 (PCL)
   ...
   ```

3. To list attributes in a section for header and trailer pipelines for the `que` queue and the `dev` device, enter:

   ```
   lsvirpt -qque -ddev -s__HTP
   ```

The output from this command is:

```
Name    Description                     Value
sh      Pipeline for Header Page        %Ide/pioburst %F[H]
                                        %Idb/H.ascii |
                                        %Ide/pioformat
                                        -@%Idd/%Imm
                                        -!%Idf/piof5202 -L! -J!
                                        %IsH
st      Pipeline for Trailer Page       %Ide/pioburst %F[H]
                                        %Idb/T.ascii |
                                        %Ide/pioformat
                                        -@%Idd/%Imm
                                        -!%Idf/piof5202-L!
                                        -t%o%G_1%r%{14}%-%d
                                        %IsT
```

4. To list all the data streams supported for the `que` queue and the `dev` device, enter:

   ```
   lsvirpt -qque -ddev -D
   ```

   The output from this command is:

   ```
   a ASCII
   p pass-through
   s PostScript
   ```

5. To list names and descriptions of all attributes in a printer attribute database for the `que` queue and the `dev` device in a specific format, enter:

   ```
   lsvirpt -qque -ddev -a'.*' -f' %3$5.5s: %6$s\\n'
   ```

   The output from this command is:

   ```
   __FLG:  Values That May Be Overridden With Flags
   _A:     stderr returned?
   _E:     Double spacing flag
   _F:     (not used) Font file name
   _H:     Name to Replace Host Name of Burst Page
   ...
   ```

6. To list all the sections in a printer attribute data base for the `que` queue and the `dev` device in a specific format, enter:

   ```
   lsvirpt -qque -ddev -a'__.*' -f'%3$s: %6$s\\n'
   ```

   The output from this command is:

   ```
   __FLG:  Values That May Be Overridden With Flags On the Command
           Line
   __SYS:  Other Values Of Interest To the Streams Administrator
   __IDS:  Pipelines For Input Data Streams (2 char,1st="i",2nd=data
           stream name)
   __PFL:  Flags Prohibited For Input Data Streams (2 char,1st="I",
           2nd=data stream name)
   __FIL:  Command Strings For Filter Flags (2 char, 1st="f",
           2nd=flag)
   __DIR:  Directories
   ...
   ```

Files

/etc/qconfig	Contains the configuration file.
/usr/sbin/lsvirprt	Contains the **lsvirprt** command.
/var/spool/lpd/pio/@local/custom/*	Contains virtual printer attribute files.
/var/spool/lpd/pio/@local/ddi/*	Contains the digested virtual printer attribute files.

Related Information

The **chvirprt** command, **mkvirprt** command, **rmvirprt** command.

The **qconfig** file.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Configuring a Printer without Adding a Queue in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Queuing System Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Virtual Printer Definitions and Attribute Values in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Adding a Printer Using the Printer Colon File in *AIX 5L Version 5.2 Guide to Printers and Printing*.

---

lsvmode Command

Purpose

Display the current video mode of the X server.

Note: This command is usable only while the X server is running.

Syntax

lsvmode

Description

The **lsvmode** command displays the current output device and viewport size used by the X server.

Security

Access Control: Any User

Auditing Events: None

Exit Status

The following exit values are returned:

0 Successful completion.
>0 An error occurred.

Examples

To display the current video mode of the X server.

```
lsvmode
```

Something similar to the following displays:

```
Current video mode information
Logical screen size   [1024x768]
Viewport size        [ 640x480]
Vertical sync. (Hz)       [60]
Active output device  [LCD][CRT]
```

Files

/usr/bin/X11/lsvmode	Contains the **lsvnode** command.

Related Information

The **chvmode** command.

---

lswlmconf Command

Purpose

Lists Workload Manager (WLM) configurations.

Syntax

lswlmconf [**-r** | **-s** | **-c** | **-d** *Config*] [**-l**] [**-t** *TimeSpec*]

Description

The **lswlmconf** command lists by default all WLM configurations and, using its flags, it is able to do the following:

- Tell which is the current configuration or set name.
- Give the list of all existing regular WLM configurations.
- Give the list of all existing WLM configuration sets.
- Tell which configuration of a set is (or would be) applicable currently (or at some time of the week).
- Tell the type of a configuration.

Flags

-c	Restricts the displayed configurations to the current configuration or set.
-d *Config*	Restricts the displayed configurations to the *Config* configuration or set.
-l	Modifies the way dates are displayed for configuration sets (ineffective for regular configurations). Sets are displayed with their currently applicable regular configuration, in the form *confset/config*.
-r	Restricts the displayed configurations to regular configurations only.
-s	Restricts the displayed configurations to configuration sets only.
-t *TimeSpec*	Uses *TimeSpec* instead of the current time to display applicable regular configuration of sets. *TimeSpec* consists in the day of the week (0 for Sunday to 6 for Saturday) and the time of the day in 24 hours format separated with a comma, in a form similar to time ranges as described in the **confsetcntrl** command. For example, to know which configuration would apply on Mondays at noon, use `-t 1,12:00`. **Note:** The **-t** flag is only effective with **-l** flag.

Examples

The following examples demonstrate how to display, change, and use WLM configurations using the **lswlmconf** command, the **confsetcntrl** command, the **wlmcheck** command, and the **wlmcntrl** command.

1. To find the WLM configurations, type:

   ```
   lswlmconf
   ```

   The output to this command might look similar to the following:

   ```
   standard
   template
   fvtrules
   fvtlimits
   fvtregul
   fvtdfct
   fvtsynt
   fvtthreads
   ```

2. To show the current WLM configuration, type:

   ```
   lswlmconf -c
   ```

   The output might look similar to the following:

   ```
   fvtlimits
   ```

3. To show configuration sets, use the **lswlmconf** with the **-s** flag as follows:

   ```
   lswlmconf -s
   ```

   Since this example configuration contains no configuration sets, this command produces a message indicating that no matching configuration was found.

4. In order to create a configuration set using ″standard″ as the default configuration, type:

   ```
   confsetcntrl -C confset1 standard
   ```

5. Now, use the **lswlmconf** command to show the new configuration set, as follows:

   ```
   lswlmconf -s
   ```

   The command now produces the following output:

   ```
   confset1
   ```

6. In order to use the ″fvtlimits″ configuration for ″confset1″ on week days (Monday through Friday) by specifying a time range, type:

   ```
   confsetcntrl -d confset1 -a fvtlimits 1-5
   ```

7. You might want this configuration only in the morning. You cannot change a time range. Instead you must remove the time range and then create a new time range.

   First, remove the old time range, as follows (confsetcntrl accepts day names, as reported by **locale day** or **locale abday** commands):

   ```
   confsetcntrl -d confset1 -r fvtlimits monday-friday
   ```

   Then create the new time range, as follows:

   ```
   confsetcntrl -d confset1 -a fvtlimits 1-5,8:00-12:00
   ```

8. In order to add another time range for using the ″fvtregul″ configuration on Sundays, type:

   ```
   confsetcntrl -d confset1 -a fvtregul 0
   ```

9. In order to display configuration set ″confset1″, type:

   ```
   confsetcntrl -d confset1
   ```

   In this example, this command produces the following output:

   ```
   fvtlimits:
           time = "1-5,8:00-12:00"

   fvtregul:
           time = "0"

   standard:
           time = "-"
   ```

10. In order to create a configuration set called ″confset2″ using ″template″ as the default configuration, type:

```
confsetcntrl -C confset2 template
```

    In order change ″confset2' so it will use the configuration ″fvtsynt″ every nigh, type:

```
confsetcntrl -d confset2 -a fvtsynt 18:00-10:00
```

11. In order to display the list of regular configurations, type:

```
lswlmconf -r
```

    In this example, this produces the following output, (which demonstrates that in this example the list of regular configurations has not changed):

```
standard
template
fvtrules
fvtlimits
fvtregul
fvtdfct
fvtsynt
fvtthreads
```

    However, as expected, the list of configurations sets in this example has changed, as shown by the following command:

```
lswlmconf -s
```

    This command produces the following output in this example:

```
confset1
confset2
```

12. In order to show which configuration would be currently active when that the **date** command reports the current time as ″Tue Jul 16 18:55:10 EET 2002″ with configuration set ″confset2″, type:

```
lswlmconf -d confset2 -l
```

    In this example, this command produces the following output:

```
confset2/fvtsynt
```

    You can also show which configurations would be active at another time. To show which configurations would be active on Sunday at 9:00am, type:

```
lswlmconf -l -t 0,9:00
```

    This command produces the following output in this example:

```
standard
template
fvtrules
fvtlimits
fvtregul
fvtdfct
fvtsynt
fvtthreads
confset1/fvtregul
confset2/fvtsynt
```

    In order to display this information only for configuration sets, type:

```
lswlmconf -s -l -t 0,9:00
```

    This produces the following output in this example:

```
confset1/fvtregul
confset2/fvtsynt
```

13. In order to remove configuration set ″confset2″, type:

```
confsetcntrl -D confset2
```

    lswlmconf -s now produces the following output in this example:

```
confset1
```

14. In order to check configuration set ″confset1″, use the **wlmcheck** command as follows:

```
wlmcheck -d confset1
```

    In this example, this produces the following output:

```
WLM is not running.
Checking classes and rules for 'confset1' configuration...
fvtlimits/System
fvtlimits/Default
fvtlimits/Shared
fvtlimits/login
fvtregul/System
fvtregul/Default
fvtregul/Shared
standard/System
standard/Default
standard/Shared
```

15. In order to start using configuration set ″confset1″ used in this example, type:

```
wlmcntrl -a -d confset1
```

    The command **lswlmconf -c** now produces the following output:

```
confset1
```

    The command **lswlmconf -cl**, which shows the active regular configuration, now produces the following output:

```
confset1/standard
```

Files

The configurations or sets files are subdirectories of **/etc/wlm**.

Related Information

The **wlmcntrl** command, **confsetcntrl** command.

---

lvmstat Command

Purpose

Reports input/output statistics for logical partitions, logical volumes and volume groups.

Syntax

lvmstat { **-l** | **-v** } *Name* [**-e** | **-d**] [**-F**] [**-C**] [**-c** *Count*] [**-s**] [*Interval* [*Iterations*]]

Description

The **lvmstat** command generates reports that can be used to change logical volume configuration to better balance the input/output load between physical disks.

By default, the statistics collection is not enabled in the system. You must use the **-e** flag to enable this feature for the logical volume or volume group in question. Enabling the statistics collection for a volume group enables for all the logical volume in that volume group.

The first report generated by **lvmstat** provides statistics concerning the time since the system was booted. Each subsequent report covers the time since the previous report. All statistics are reported each time **lvmstat** runs. The report consists of a header row followed by a line of statistics for each logical partition or logical volume depending on the flags specified.

If the **-l** flag is specified, *Name* is the logical volume name, and the statistics are for the physical partitions of this logical volume. The mirror copies of the logical partitions are considered individually for the statistics reporting. They are listed in descending order of number of i/os (iocnt) to the partition.

The *Interval* parameter specifies the amount of time, in seconds, between each report. The first report contains statistics for the time since the volume group startup, **varyonvg**. Each subsequent report contains statistics collected during the interval since the previous report. If the *Count* parameter is specified, only the top *Count* lines of the report are generated. For a logical volume if *Count* is 10, only the 10 busiest partitions are identified. If the *Iterations* parameter is specified in conjunction with the *Interval* parameter, then only that many iterations are run. If no *Iterations* parameter is specified, **lvmstat** generates reports continuously. If *Interval* is used to run lvmstat more than once, no reports are printed if the statistics did not change since the last run. A single period **.** (period) is printed instead.

The **lvmstat** command is useful in determining whether a physical volume is becoming a hindrance to performance by identifying the busiest physical partitions for a logical volume.

Note: The **lvmstat** commands reports I/O statistics of the local node only.

Reports

The **lvmstat** command generates two types of reports, per partition statistics in a logical volume and per logical volume statistics in a volume group. The reports has the following format:

Column	Description
Log_part	Logical partition number
mirror#Log_part	Mirror copy number of the logical partition
iocntLog_part	Number of read and write requests
Kb_readLog_part	The total number of kilobytes read
Kb_wrtnLog_part	The total number of kilobytes written
KbpsLog_part	The amount of data transferred in kilobytes per second

Flags

-c *Count*	Prints only the specified number of lines of statistics.
-C	Causes the counters that keep track of the iocnt, Kb_read and Kb_wrtn be cleared for the specified logical volume/volume group.
-d	Specifies that statistics collection should be disabled for the logical volume/volume group in question.
-e	Specifies that statistics collection should be enabled for the logical volume/volume group in question.
-F	Causes the statistics to be printed colon-separated.
-l	Specifies the name of the stanza to list.
-s	Suppresses the header from the subsequent reports when *Interval* is used.
-v	Specifies that the *Name* specified is the name of the volume group.

Security

To use **lvmstat**, you must have root user authority.

Examples

1. To enable the statistics collection for volume group datavg (all the LVs in datavg are enabled), type:

   ```
   lvmstat -v datavg -e
   ```

2. To display the history of all the partitions of logical volume hd2, type:
   ```
   lvmstat -l hd2
   ```
3. To display the history of top five logical volumes of volume group uservg, type:
   ```
   lvmstat -v uservg -c 5
   ```
4. To display a continuous report at two second intervals for the logical volume ramlv, type:
   ```
   lvmstat -l ramlv 2
   ```
5. To display six reports at two second intervals for the volume group rootvg, type:
   ```
   lvmstat -v rootvg 2 6
   ```
6. To reset the counters for statistics for all the logical volumes in the volume group uservg, type:
   ```
   lvmstat -v uservg -C
   ```
7. To disable statistics collection for datalv, type:
   ```
   lvmstat -l datalv -d
   ```

Files

/usr/sbin/lvmstat	Contains the **lvmstat** command.

Related Information

The **knlist** subroutine.

The **/dev/kmem** special file.

---

m4 Command

Purpose

Preprocesses files, expanding macro definitions.

Syntax

m4 [**-e**] [**-l**] [**-s**] [**-B** *Number*] [**-D** *Name* [**=***Value*]] ... [**-H** *Number*] [**-I** *Directory*] [**-S** *Number*] [**-T** *Number*] [**-U** *Name*] ... [*File* ...]

Description

The **m4** command is a macro processor used as a preprocessor for C and other languages. You can use it to process built-in macros or user-defined macros.

Each *File* parameter is processed in order. If you do not specify a *File* parameter or if you specify the **-** (dash) as a file name, the **m4** command reads standard input. It writes the processed macros to standard output. Macro calls follow the form:

```
macroname(argument . . . )
```

The left parenthesis must immediately follow *macroname*. If the left parenthesis does not follow the name of a defined macro, the **m4** command reads it as a macro call with no arguments. Macro names consist of ASCII alphabetic letters, digits, and the _ (underscore) character. Extended characters are not allowed in macro names. The first character cannot be a digit.

While collecting arguments, the **m4** command ignores unquoted leading blanks, tabs, and new-line characters. Use single quotation marks to quote strings. The value of a quoted string is the string with the quotation marks stripped off.

When the **m4** command recognizes a macro, it collects arguments by searching for a matching right parenthesis. If you supply fewer arguments than appear in the macro definition, the **m4** command considers the trailing arguments in the definition to be null. Macro evaluation proceeds normally during the collection of the arguments. All commas or right parentheses within the value of a nested call are translated literally; they do not need an escape character or quotation marks. After collecting arguments, the **m4** command pushes the value of the macro back onto the input stream and scans again.

Built-in Macros

The **m4** command makes available the following built-in macros. You may redefine them, but you will lose the original meaning. The values of these macros are null unless otherwise stated:

define(*Name*,*NewName*)	Replaces the macro *Name* with the value of *NewName*. The *NewName* string can take the form $*n* . . . (where *n* is a digit). In this case, each occurrence of *n* in the replacement text is replaced by the *n*th argument of *Name*. $0 is the name of the macro. The null string replaces missing arguments. The number of arguments replaces **$#**. A comma-separated list of all arguments replaces $*. $@ acts like $*, but each argument is quoted with the current quotation character (see **changequote**).
undefine(*Name*)	Removes the definition of *Name*.
defn(*Name* . . .)	Returns the quoted definition of *Name*.
pushdef(*Name***,** *NewName***)**	Redefines *Name* with *NewName* as in **define**, but saves any previous definition.
popdef(*Name* . . .)	Removes the current definition of *Name* and returns to the previous definition, if one existed.
ifdef(*Name*,*True*,[*False*])	Returns the value of *True* only if *Name* is defined and is not defined to be 0, otherwise returns *False*. If you do not supply *False*, its value is null.
shift(*Argument* . . .)	Returns all but the first argument. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that is subsequently performed.
changequote(*L*,*R*)	Changes quote symbols to *L* and *R*. The symbols can be up to 5 bytes long. **changequote** without arguments restores the original values (` ').
changecom(*L*,*R*)	Changes left and right comment markers from the default # and new-line character to *L* and *R*. With no arguments, the comment mechanism is disabled. With one argument, the left marker becomes the parameter and the right marker becomes a new-line character. With two arguments, both markers are affected. Comment markers can be up to 5 bytes long.
divert(*Number*)	Changes the current output stream to stream *Number*. There are 10 output streams, numbered 0-9. The final output is the concatenation of the streams in numerical order. Initially, stream 0 is the current stream. The **m4** command discards output diverted to a stream other than 0-9.
undivert(*Number* . . .)	Causes immediate output of text from the specified diversions (or all diversions if there is no argument). Text may be undiverted into another diversion. Undiverting discards the diverted text.
divnum	Returns the value of the current output stream.
dnl	Reads and discards characters up to and including the next new-line character.
ifelse([*String1***,***String2***,***True***,**[*False*]] . . .	If)*String1* and *String2* are the same then the value is *True*. If they are not and if there are more than four arguments, the **m4** command repeats the process with the additional arguments (4, 5, 6, and 7). Otherwise, the value is either *False* or null if you provide no value for *False*.
incr(*Number*)	Returns the value of its argument incremented by 1.
decr(*Number*)	Returns the value of its argument decreased by 1.
eval(*Expression*[,*Number1*[,*Number2*]])	Evaluates its first argument as an arithmetic expression, using 32-bit arithmetic. The operators you can use are +, -, *, /,%, ^ (exponentiation), bitwise &, | , ~, and ^ relationals, and parentheses. Octal and hex numbers can be specified as in C. *Number1* specifies the radix for the result of the expression. The default radix is 10. The optional *Number2* specifies the minimum number of digits in the result.

len(*String*)	Returns the number of bytes in *String*.
dlen(*String*)	Returns the number of displayable characters in *String*; that is, two-byte extended characters are counted as one displayable character.
index(*String1*,*String2*)	Returns the position in the *String1* string where the *String2* string begins (zero origin), or -1 if the second parameter does not occur.
substr(*String*,*Position*, [*Number*])	Returns a substring of *String*. The beginning of the substring is selected with *Position*, and *Number* indicates the length of the substring. Without *Number*, the substring includes everything to the end of the first string.
translit(*String*,*From*,*To*)	Transliterates the characters in *String* from the set given by *From* to the set given by *To*. No abbreviations are permitted. Two-byte extended characters are correctly mapped into the corresponding replacement characters.
include(*File*)	Returns the contents of *File* or displays an error message if it cannot access the file.
sinclude(*File*)	Returns the contents of *File*, but it gives no error message if *File* is inaccessible.
syscmd(*Command*)	Runs the *Command*. No value is returned.
sysval	Returns the return code from the last call to **syscmd**.
maketemp(. . . *nnnn* . . .)	Replaces *nnnn* in its argument with the current process ID number.
m4exit(*Value*)	Exits from **m4** immediately, returning the specified exit *Value* (the default is 0).
m4wrap(*LastMacro*)	Runs *LastMacro* after reading the end-of-file character. For example, `m4wrap (`cleanup ()')` runs the cleanup macro at the end of **m4**.
errprint(*Message*)	Includes *Message* on the diagnostic output file.
dumpdef([*Name* . . .])	Writes to standard output the current names and definitions for the named items or for all if no arguments are provided.
traceon(*Macro*)	Turns on tracing for *Macro*. If none is named, tracing is turned on for all macros.
traceoff(*Macro* . . .)	Turns off trace globally and for any *Macro* specified. Macros specifically traced by **traceon** can be untraced only by specific calls to **traceoff**.

Flags

-B *Number*	Makes the *Number* variable the size of the push-back and parameter collection buffers (the default is 4096).
-e	Operates interactively. Interrupts are ignored and the output is not buffered.
-H *Number*	Makes the *Number* variable the size of the symbol table hash array (the default is 199). The size must be a prime number.
-I *Directory*	(Uppercase i) Searches the *Directory* variable first, then searches the directories on the standard list for include (built-in macro) files with names that do not begin with a **/** (slash).
-l	(Lowercase L) Enables line-numbering output for the assembler (.xline . . .).
-s	Enables the line-sync output for the C preprocessor (#line . . .).
-S *Number*	Makes the *Number* variable the size of the call stack (the default is 100 slots). Macros take three slots, and non-macro arguments take one.
-T *Number*	Makes the *Number* variable the size of the token buffer (the default is 512 bytes).

The preceding flags must appear before any file names and before any **-D** or **-U** flags.

-D *Name*[**=***Value*]	Defines the *Name* variable as the *Value* variable. If the *Value* variable is not specified, the *Name* variable becomes null.
-U *Name*	Undefines a the *Name* variable previously defined with the **-D** flag.

Exit Status

This command returns the following exit values:

0	Successful completion.

>0 An error occurred.

If the **m4exit** macro is used, the exit value can be specified by the input file.

Examples

To preprocess a C language program with the **m4** command and compile it, enter:

```
m4 prog.m4 > prog.c
cc prog.c
```

Files

/usr/ccs/bin/m4 Contains the **m4** command.

Related Information

The m4 Macro Processor Overview in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

The Commands Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

The **as** command, **cc** command, **cpp** command.

---

mach Command

Purpose

Displays the processor type of the current host .

Syntax

mach

Description

The **mach** command displays the architecture of the system processor.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Examples

1. To display the processor type of the current host use the **mach** command in the following way:

   ```
   mach
   ```

Files

/usr/bin/mach Contains the System V **mach** command.

Related Information

The **uname** command.

machstat Command

Purpose
Reports the value of the first 4 bits of the power status register.

Syntax
machstat -p

Description
The **machstat** command returns the value of a status register. There is no standard output or error.

Flags

-p Displays the first 4 bits of the power status register.

Exit Status
The **machstat** command returns a value of 255 if an error occurs. Otherwise it returns the value of the register.

Security
Access Control: root only

Examples
To see the current value of the power status register, enter:

```
machstat -p
echo $?
```

Files

/etc/rc.powerfail	Shuts down a system when a power failure is detected

Related Information
The **rc.powerfail** command.

macref Command

Purpose
Produces a cross-reference listing of macro files.

Syntax
macref [**-n**] [**-s**] [**-t**] [**—**] [*File* ...]

Description
The **macref** command reads the named English-language files (which are assumed to be **nroff** or **troff** command input) and produces a cross-referenced listing of the symbols in the input.

The default output is a list of the symbols found in the input, each accompanied by a list of all references to that symbol. The **macref** command lists the symbols alphabetically in the left column, with references following to the right. Each reference is given in the following form:

```
[ [(  NMName ) ]
 MName- ]
 Type  LNumber
[  # ]
```

Generated names are listed under the artificial symbol name ~sym.

Input Parameters

File Specifies the **nroff** or **troff** file from which the **macref** command produces output containing a list cross-referencing macros.

Output Parameters

NMName The name of the macro within which MName is defined.

MName The name of the macro within which the reference occurs. This field is not present if the reference occurs outside a macro.

Type The type associated, by context, with this occurrence of the symbol. The types can be the following:

r Request

m Macro

d Diversion

s String

n Number register

p Parameter. For instance, \$x is a parameter reference to x.

Note: Parameters are never modified, and the only valid parameter symbol names are 1, 2, . . . 9.

LNumber The line number on which the reference occurred.

This reference modifies the value of the symbol.

Flags

-n Causes one line to be printed for each reference to a symbol.

-s Causes symbol-use statistics to be printed.

-t Causes a macro table of contents to be printed.

The flags can be grouped behind one - (minus sign). Use a — (dash) to delimit the end of flags.

Note: The **macref** command does not accept - as standard input.

Files

/tmp/macref.t*XXXXXX* Contains a temporary file.

/tmp/macref.s*XXXXXX* Contains a temporary file.

/tmp/macref.c*XXXXXX* Contains a temporary file.

Related Information

The **mm** command, **mmt** command, **mvt** command, **nroff** command, **troff** command.

The **man** macro package, **mm** macro package, **mv** macro package.

---

mail, Mail, or mailx Command

Purpose

Sends and receives mail.

Syntax

To Read Incoming Mail

mail -e

mail -f [**-dHNn**] [**-F**] [*FileName*]

mail [**-dHNn**] [**-F**] [**-u** *UserID*]

To Send Mail

mail [**-s** *Subject*] [**-c** *Address(es)*] [**-dinNv**] *Address*

Description

The **mail** command invokes the mail utility, enabling you to:

- Read incoming mail.
- Send mail.

In addition, you can use the available options and subcommands to customize the way you send and receive mail.

The **mail** command operates on two types of mailboxes, the system mailbox and the personal mailbox.

Incoming mail is stored in the system mailbox. By default, a user's system mailbox is a file located in the /**var**/**spool**/**mail** directory. The mailbox file is named after the userID. For example, if your userID is *jeanne*, then your system mailbox is /**var**/**spool**/**mail**/**jeanne**.

By default, when a user has read, deleted, or saved all the mail in their system mailbox, the mailbox is deleted. To prevent the mailbox from being deleted, use the **set** subcommand to set the **keep** option.

In addition to the system mailbox, there is the user's personal mailbox. As messages are read, if they are not deleted or saved to a file, they will be marked to be moved to the personal mailbox. The personal mailbox, by default, is **$HOME/mbox**. For example, if your home directory is **/home/lance**, then **/home/lance/mbox** is your personal mailbox. The messages remain in your personal mailbox until you move them to a folder or delete them.

Folders provide a way to save messages in an organized fashion. You can create as many folders as you need. Name each folder with a name that pertains to the subject matter of the messages it contains.

Note: Results can be unpredictable when running multiple instances of the **mail** command on one mailbox.

Examining the Contents of Your Mailbox

To process your mail, type `mail` at the system prompt. The Mail program displays a one-line entry for each piece of mail in your system mailbox:

```
Mail [5.2 UCB] [AIX 4.1] Type ? for help.
"/var/spool/mail/lance": 2 messages 2 new
>N 1 karen         Thu Sep 17 14:36 13/359 "Dept Meeting"
 N 2 lance@zeus    Thu Sep 17 15:06 10/350 "Delay"
 N 3 karen         Thu Sep 17 14:36 13/359 "Meeting Cancel"
```

The current message is marked by a > at the beginning of a line in the header summary.

Each one-line entry displays the following fields:

status Indicates the current class of a piece of mail. The status can be any of the following:

- **N** A new message
- **P** A message to be preserved in system mailbox.
- **U** An unread message. An unread message is a message that was listed in the mailbox last time you invoked the Mail program, but whose contents you did not examine.
- ***** A message that was saved or written to a file or folder.

A message without a status indicates that the message has been read but has not been deleted or saved.

number Identifies the numerical order of the message.
sender Identifies the address of the person who sent the mail.
date Specifies the date the message was received.
size Defines the number of lines and characters contained in the letter (this includes the header).
subject Identifies the subject of the message.

Finally, following the list of mail, the Mail program displays the mailbox prompt, which by default is **?**, to indicate that it is waiting for input.

Flags

-c *Address(es)* Specifies the list of users to which a copy of the message is sent. You can specify one or more addresses. When specifying more than one address, the list of addresses must be in (″ ″) quotes.
-e Tests for the presence of mail in the system mailbox. The **mail** utility will write nothing and exit with a successful return code if there is mail to read.
-f *FileName* Reads messages from the named file. If a file operand is not specified, then reads messages from **mbox**. When you quit from reading the messages, undeleted messages are written back to this file.
-F Records the message in a file named after the recipient. The name is the portion of the address found first on the **To:** line in the mail header. Overrides the **record** variable if set.
-H Writes a header summary only.
-i Causes tty interrupt signals to be ignored.
-n Inhibits reading the **/usr/share/lib/Mail.rc** file.
-N Suppresses the initial printing of headers.
-s *Subject* Specifies a subject for a message to be created.
-u *UserID* Specifies an abbreviated equivalent of doing **mail -f /var/spool/mail/***UserID*. Starts the Mail program for a specified user's mailbox. You must have access permission to the specified mailbox.
-v Puts the Mail program into verbose mode. Displays the details of delivery on the user's terminal.

Environmental Variables

The following environment variables affect the execution of mail:

DEAD Pathname of the file in which to save partial messages in case of interrupts or delivery errors.

EDITOR Pathname of the editor to use when the **edit** or **~e** command is used.

HOME Pathname of the user's home directory.

LISTER String representing the command for writing the contents of the folder directory to standard output when the **folders** command is given. Any string acceptable as a command_string operand to the **sh -c** command is valid. If this variable is null or not set, the output command will be *ls*. The default value is unset.

MAILBOX Specifies the location of the system mailbox for the **mail** command. The **MAILBOX** value is where the **mail** command searches for mail messages. The system default value if the **MAILBOX** environment variable is not specified is the **/var/spool/mail** directory.

MAILRC Pathname of your personal startup file. The default is **$HOME/.mailrc**.

MBOX Pathname of your personal mailbox where messages are saved from the system mailbox that have been read. The **exit** command overrides this function, as will saving the message explicitly in another file. The default is **$HOME/mbox**.

PAGER String representing an output filtering or pagination command for writing the output to the terminal. Any string acceptable as a command_string operand to the **sh-c** command is valid. When standard output is a terminal device, the message output will be piped through the command if the mail internal variable **crt** is set to a value less the number of lines in the message. If the **PAGER** variable is null or not set, the paginator is the **pg** shell command.

SHELL Pathname of a preferred command interpreter.

VISUAL Pathname of a utility to invoke when the **visual** command or **~v** command-escape is used. If this variable is not set, the full screen editor will be *vi*.

Internal Variables in Mail

allnet Treats all network names, whose login name components match, identically. Causes the **msglist** message specifications to behave similarly. The default is **noallnet**.

append Adds the message saved in your mailbox to the end, rather than the beginning, of the **$HOME/mbox** file. The default is **noappend**.

ask, asksub Prompts for the subject of each message if not specified on the command line with the **-s** option. If you do not wish to create a subject field, press Enter at the prompt. It is not possible to set both **ask** and **noasksub**, or **noask** and **asksub**. The default is **asksub**.

askbcc Prompts for the addresses of people to add to the blind copy list. If you do not wish to send blind copies, press Enter at the prompt.

askcc Prompts for the addresses of people who should receive copies of the message. If you do not wish to send copies, press Enter at the prompt.

autoprint Sets the **delete** subcommand to delete the current message and display the next message.

crt Specifies the minimum number of lines that a message must contain before any output filtering or pagination is used when the message is displayed.

debug Displays debugging information. Messages are not sent while in debug mode. This is the same as specifying the **-d** flag on the command line.

dot Interprets a period entered on a line by itself as the end of a message you are sending.

escape=*c* Sets the command escape character to be the character *c*. By default the command escape character is ~ (tilde).

Replyall, flipr Reverses the meanings of the **Respond** and **respond** or **Reply** and **reply** commands. The default is **noflipr**.

folder=*directory* The directory name in which to store mail folders. After the directory is defined, you can use the + (plus sign) notation to refer to it when using the *FileName* parameter with **mailbox** subcommands.

header	Enables writing of the header summary when entering mail in receive mode. The default is **header**.
hold	Holds messages that you have read but have not deleted or saved in the system mailbox instead of in your personal mailbox. The default is **nohold**.
ignore	Ignores interrupts while entering messages. Echoes interrupts as @ (at) characters.
ignoreeof	Sets the **mail** command to refuse the Ctrl+D key sequence as the end of a message. Input can be terminated only by entering a period . (period) on a line by itself or by the **~.** command escape. The default is **noignoreeof**.
indentprefix=*string*	A string that will be prefixed to each line that is inserted into the message by the **~m** command escape. This variable defaults to one tab character.
keep	When a system mailbox, secondary mailbox, or mbox is empty, truncate it to zero length instead of removing it. The default is **nokeep**.
keepsave	Keep messages that have been saved with the **(s)ave** or **(w)rite** subcommands in the system mailbox instead of deleting them. The default is **nokeepsave**.
metoo	Includes the sender in the alias expansion if sender's name is part of the alias. By default, expanding the alias removes the sender.
onehop	When responding to a message that was originally sent to several recipients, the other recipient addresses are usually forced to be relative to the originating author's machine for the response. This flag disables alteration of the recipient's addresses, improving efficiency in a network where all machines can send directly to all other machines (that is, one hop away). The default is **noonehop**.
outfolder	Causes the files used to record outgoing messages to be located in the directory specified by the **folder** variable unless the pathname is absolute. The default is **nooutfolder**. See the **record** and **folder** variables.
page	Insert a form-feed after each message sent through the pipe created by the **pipe** command. The default is **nopage**.
prompt=*string*	Set the command-mode prompt to *string*. If *string* is null or if **noprompt** is set, no prompting will occur. The default is to prompt with the ″?″ string.
quiet	Refrain from writing the opening message and version when entering mail. The default is **noquiet**.
record=*file*	Defines a file in which to record all outgoing mail. The default is **norecord**.
save	Enables saving of messages in the **dead.letter** file on interrupt or delivery error. The default is **save**.
screen=*number*	Sets the number of lines in a screenful of headers for the **headers** and **z** commands.
sendmail=*shell_command*	Alternative command for delivering messages.
sendwait	Wait for the background mailer to finish before returning. The default is **nosendwait**.
showto	When the sender of the message was the user who is invoking mail, write the information from the **To:** line instead of the **From:** line in the header summary. The default is **noshowto**.
sign=*string*	Inserts *string* into the text of a message when the **~a** command escape is given. The default is **nosign**. The character sequences **/t** and **/n** are recognized in the string as tab and newline characters, respectively.
Sign=string	Inserts *string* into the text of a message when the **~A** command escape is given. The default is **noSign**.
toplines=number	Number of lines displayed by the **top** subcommand.
verbose	Displays the actual delivery of messages on the terminal. This is the same as specifying the **-v** flag on the command line.

Setting Environment Variables

The Bourne shell (**bsh** command) uses and checks the following variables. These variables can be set in **$HOME/.profile**.

MAIL Specifies the location and name of the user's system mailbox that is checked by the Bourne shell to determine whether or not you have mail. If the system mailbox is not empty, the Bourne shell sends a message that you have new mail. The Bourne shell checks the system mailbox periodically based on the value of the **MAILCHECK** environment variable.

MAILCHECK Specifies the interval at which the Bourne shell checks the system mailbox for mail.

MAILMSG Specifies the message sent to your console shell by the system when you have mail. The default message is similar to the following:

```
YOU HAVE NEW MAIL
```

Examples

1. To start the Mail program and list the messages in your mailbox, type the following at the command line prompt:

   ```
   mail
   ```

   The **mail** command lists every messages in your system mailbox. The mail system then displays the mailbox prompt (?) to indicate *waiting for input*. When you see this prompt, enter any mailbox subcommand. To see a list of subcommands, type:

   ```
   ?
   ```

   This entry lists the Mail subcommands.

2. To send the message `letter` to the recipient `user1@host1` and copies to `user2@host2` and `user3@host3`, type:

   ```
   mail -c "user2@host2 user3@host3" user1@host1<letter
   ```

3. To look at the contents of your personal mailbox, type:

   ```
   mail -f
   ```

   This command displays a list of the messages in your personal mailbox, **$HOME/mbox**.

4. To look at the contents of a specific mail folder, type:

   ```
   mail -f +dept
   ```

   This command displays a listing of the messages in the `dept` folder.

5. To send a message to a user on your local system, type:

   ```
   mail ron
   ```

   When you finish typing the message to user `ron`, press Enter and press either . (period) or Ctrl+D to exit the editor and send the message. To determine if a user is on your local system, check for the user's name in your **/etc/passwd** file.

   If your message is delivered successfully, you receive no notification. If your message could not be delivered, an error message is sent to you.

6. To mail a file to another user on your local system, type:

   ```
   mail karen < letter1
   ```

   This command sends the contents of the file `letter1` to user `karen` on your local system. After the command sends the file, the Mail program displays the command line prompt.

7. To send a message to a user on a remote system, type:

   ```
   mail dale@zeus
   ```

You now can create a message to `dale`. In this example, you are sending a message to user `dale` on remote system `zeus`. To send a message to a user on another system connected to your system through a network, you must know that person's login ID and the name of the other system.

Mailbox Subcommands for the mail, Mail, and mailx Command

From the mail prompt, ? (question mark), you can enter subcommands to manipulate mail in your mailbox. Subcommands that work on more than one message at a time use the *MessageList* parameter. Subcommands that work with files or folders use the *FileName* parameter. These parameters are discussed in "Summary of Mailbox Subcommands in the Mail Program" in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

The following list describes the Mailbox subcommands and their functions:

= Echoes the number of the current message.

Comment character for writing comments in mail script files.

- Displays the previous message.

? Displays a brief summary of mailbox subcommands. Identical to the **help** subcommand.

!*Command* Executes the workstation shell command specified by *Command*.

alias (**a**) With no arguments, displays all currently defined aliases and their corresponding addresses. With one argument, displays one alias. With more than one argument, creates a new alias or changes an old alias. Identical to the **group** subcommand.

alternates *AlternatesList* (**alt**) The **alternates** subcommand is useful if you have accounts on several machines. Use the subcommand to inform the Mail program that the addresses listed in *AlternatesList* all refer to you. When you use the **reply** subcommand to reply to messages, the Mail program does not send a copy of the message to any of the addresses given in *AlternatesList*. If you enter the **alternates** subcommand with no argument, the Mail program displays the current set of alternate names.

chdir *Directory* (**cd**) Changes your working directory to the indicated *Directory*. If no directory is given, it changes to your login directory.

copy [*MessageList*] *File* (**c**, **co**) Appends each message in *MessageList* to the end of *File*. Displays the file name in quotes, followed by the line count and character count, on the user's terminal. Does not delete any messages when you quit.

Copy [*MessageList*] (**C**) Saves the specified message in a file whose name is derived from the author of the message to be saved, without marking the messages as saved. Otherwise, it is equivalent to the **Save** subcommand.

delete [*MessageList*] (**d**) Marks the messages in *MessageList* to be deleted when you quit the Mail program. Entering the **d** subcommand without a message list deletes the current message. Deleted messages are not saved in your **$HOME/mbox** file nor are they available for most other commands. However, you can use the **undelete** subcommand to restore messages you have deleted while in the same mailbox session. If you delete a message and either change to another mailbox or quit the mailbox with the **quit** subcommand, the deleted message cannot be recalled.

discard [*FieldList*] (**di**) Identical to the **ignore** subcommand.
Note: The **retain** subcommand overrides the **discard** subcommand.

dp Deletes the current message and displays the next message. If there is no next message, the Mail program displays `EOF`. Identical to the **dt** subcommand.

dt Deletes the current message and displays the next message. If there is no next message, the Mail program displays `EOF`. Identical to the **dp** subcommand.

echo *String* Displays the character string *String* on the command line.

edit [*MessageList*] (**e**) Starts the alternate editor using the *MessageList* as input files. To define an alternate editor, use the **set EDITOR=** statement or place an entry in your **$HOME/.mailrc** file. Any message specified by the *MessageList* parameter retains the changes made during the editor session.

exit (**ex** or **x**) Leaves the mailbox and returns to the operating system without changing the original contents of the mailbox. The mailbox returns to the condition that it was when the Mail program was started. Messages marked to be deleted are not deleted. Identical to the **xit** subcommand.

file [*Name*] (**fi**) Identical to the **folder** subcommand.

folder [*Name*] (**fo**) Switches to a new mail file or folder. With no arguments, the subcommand displays the name of the current mailbox. If an argument is included, it stores the current mailbox with changes (such as messages deleted) and reads in the new mailbox specified by the *Name* parameter. Identical to the **file** subcommand.

Some special conventions are recognized for the *Name*:

`#` Refers to the previous file.

`%` Refers to the system mailbox (**/var/spool/mail/***UserID*).

& Refers to your personal mailbox (**$HOME/mbox**).

+*Name* Refers to a file in your folder directory.

folders Lists the names of the folders in your folder directory.

followup [*message*] (**fo**) Responds to a message, recording the response in a file whose name is derived from the author of the message. Overrides the **record** variable, if set.

Followup [*MessageList*] (**F**) Responds to the first message in the *msglist*, sending the message to the author of each message in the *msglist*. The subject line is taken from the first message and the response is recorded in a file whose name is derived from the author of the first message.

from [*MessageList*] (**f**) Displays the headings of messages in *MessageList*.

group (**g**) Identical to the **alias** subcommand.

headers [*Message*] (**h**) Lists the headings in the current group of messages (each group of messages contains 20 messages by default; change this with the **set screen=** statement). If the mailbox contains more messages than can be displayed on the screen at one time, information about only the first group of messages will be displayed. To see information about the rest of the messages, use the **h** subcommand with a message number in the next range of messages, or use the **z** subcommand to change the current message group.

help Displays a brief summary of mailbox subcommands. Identical to the **?** subcommand.

hold [*MessageList*] (**ho**) Marks each message in *MessageList* to be saved in your system mailbox (**/var/spool/mail/***UserID*) instead of in your **$HOME/mbox** file. Does not override the **delete** subcommand. Identical to the **preserve** subcommand.

if *Condition*

else

endif Construction for conditional execution of the **mail** subcommands. Subcommands following **if** are executed if *Condition* is true. Subcommands following **else** are executed if *Condition* is not true. The **else** is not required. The **endif** ends the construction and is required. The *Condition* can be receive (receiving mail) or send (sending mail).

ignore [*FieldList***]**
Adds the header fields in *FieldList* to the list of fields to be ignored. Ignored fields are not displayed when you look at a message with either the **type** or **print** subcommand. Use this subcommand to suppress machine-generated header fields. Use either the **Type** or **Print** subcommand to print a message in its entirety, including ignored fields. The **ignore** subcommand with no arguments lists all header fields that are not included when you use a **type** or **print** subcommand to display a message. Identical to the **discard** subcommand.

list (**l**) Displays a list of all mailbox subcommands with no explanation of what they do.

mail *AddressList*
(**m**) Starts the mail editor. Enables you to create and send a message to people specified in *AddressList*. The newly created message is independent from any receive messages.

mbox [*MessageList***]**
Indicates that the messages in *MessageList* are to be sent to your personal mailbox (**$HOME/mbox**) when you quit the Mail program. This operation is the default action for messages that you have read if you are looking at your system mailbox (**/var/spool/mail/***UserID*) and the **hold** option is not set.

more [*MessageList***]**
(**mo**) Displays the messages in *MessageList* using the defined pager program to control display to the screen. Identical to the **page** subcommand.

More [*MessageList***]**
(**Mo**) Similar to the **more** subcommand, but also displays ignored header fields.

new [*MessageList***]**
Marks each message in *MessageList* as *not* having been read. Identical to the **New**, **unread**, and **Unread** subcommands.

New [*MessageList***]**
Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **unread**, and **Unread** subcommands.

next [*Message***]**
(**n**) Makes the next message in the mailbox the current message and displays that message. With an argument list, it displays the next matching message.

page [*MessageList***]**
(**pa**) Displays the messages in *MessageList* using the defined pager program to control display to the screen. Identical to the **more** subcommand.

Page [*MessageList***]**
(**Pa**) Similar to the **page** subcommand but also displays ignored header fields.

pipe [[msglist command]]

| [[msglist] command] (**pi**) Pipe the messages through the given command by invoking the command interpreter specified by **SHELL** with two arguments: **-c** and *command*. The command must be given as a single argument. This can be accomplished by quoting. If no arguments are given, the current message will be piped through the command specified by the value of the **cmd** variable. If the **page** variable is set, a form-feed character will be inserted after each message.

preserve (**pre**) Identical to the **hold** subcommand.

print [*MessageList*]
(**p**) Displays the text of a specific message. Identical to the **type** subcommand.

Print [*MessageList*]
(**P**) Displays the text of a specific message along with the ignored header fields. Identical to the **Type** subcommand.

quit
(**q**) Leaves the mailbox and returns to the operating system. All messages read, but not deleted or saved are stored in your personal mailbox (**$HOME/mbox**). All messages you have marked to be deleted are removed from the mailbox and cannot be recovered. All messages marked with the **hold** or **preserve** option and messages you have not viewed are saved in the system mailbox (**/var/spool/mail/***UserID*). If the **quit** subcommand is given while editing a mailbox file with the **-f** flag, the edit file is saved with changes. If the edit file cannot be saved, the Mail program does not exit. Use the **exit** subcommand to exit without saving the changes.

reply [*Message*]
(**r**) Allows you to reply to the sender of a message and to all others who receive copies of the message. Identical to the **respond** subcommand.

Reply [*Message*]
(**R**) Allows you to reply to only the sender of a message. Identical to the **Respond** subcommand.

respond [*Message*]
Allows you to reply to the sender of a message and to all others who receive copies of a message. Identical to the **reply** subcommand.

Respond [*Message*]
Allows you to reply to only the sender of a message. Identical to the **Reply** subcommand.

retain [*FieldList*]
Adds the header fields in *FieldList* to the list of fields to be retained. Retained fields are displayed when you look at a message with the **type** subcommand or **print** subcommand. Use this subcommand to define which header fields you want displayed. Use the **Type** or **Print** subcommand to print a message in its entirety, including fields that are not retained. If the **retain** subcommand is executed with no arguments, it lists the current set of retained fields.
Note: The **retain** subcommand overrides the **discard** subcommand.

save [*File*]
(**s**) Saves the current message including header information to a file or folder. If the file already exists, the message is appended to the file. If *File* is omitted, the message will be saved to the user's **mbox**.

save [*MessageList*] *File*
(**s**) Saves a *MessageList* including heading information to a file or folder. If the file already exists, the *MessageList* is appended to the file. Displays the file name and the size of the file when the operation is complete. If you save a message to a file, that message is not returned to the system mailbox (**/var/spool/mail/***UserID*) nor saved in your personal mailbox (**$HOME/mbox**) when you quit the Mail program.

Save [*MessageList*]
(**S**) Saves the specified messages in a file whose name is derived from the author of the first message. The name of the file is taken to be the author's name with all network addressing stripped off.

set [*OptionList* | *Option*=*Value*...]
(**se**) With no arguments, displays the options that are currently enabled. Otherwise, sets an option as specified. The argument following the **set** command can be either:

- An *OptionList* giving the name of a binary option (an option that is either set or unset)
- An *Option*=*Value* entry used to assign a value to an option.

The options are listed in the **.mailrc** file format.
Note: The form **unset name** is equivalent to **noname**.

shell
(**sh**) Starts an interactive version of the shell.

size [*MessageList*]
Displays the sizes in lines/characters of the messages in *MessageList*.

source *File*
(**so**) Reads and executes the **mail** subcommands from *File*.

top [*MessageList*]
Displays the top few lines of the messages specified by *MessageList*. The number of lines displayed is determined by the valued option **toplines** and defaults to five.

touch [*MessageList*]
Within your system mailbox (**/var/spool/mail/***UserID*), this subcommand marks the messages in *MessageList* to be moved to your personal mailbox (**$HOME/mbox**) when you quit the Mail program. The messages are moved even though you have not read them. The messages are displayed in your personal mailbox as unread messages. The last message in *MessageList* becomes the current message.

type [*MessageList*]
(**t**) Displays the text of a specific message. Identical to the **print** subcommand.

Type [*MessageList*]
(**T**) Displays the text of a specific message along with the ignored header fields. Identical to the **Print** subcommand.

unalias
Deletes the specified alias names.

undelete [*MessageList*]
(**u**) Removes the messages in *MessageList* from the list of messages to be deleted when you quit the Mail program. Entering the **u** subcommand without a message list recalls the last deleted message.

unread [*MessageList*]
(**U**) Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **New**, and **Unread** subcommands.

Unread [*MessageList*]
Marks each message in *MessageList* as *not* having been read. Identical to the **new**, **New**, and **unread** subcommands.

unset *OptionList*
Disables the values of the options specified in *OptionList*. This action is the inverse of the **set** subcommand.
Note: The form **unset name** is equivalent to **noname**.

version	(**ve**) Displays the version banner for the Mail program.
visual [*MessageList*]	(**v**) Starts the visual editor using the *MessageList* as the input field. (This editor can be defined with the **set VISUAL=** statement.) Any changes made during the editor session are saved back to the messages in the *MessageList*.
write [*MessageList*] *File*	(**w**) Saves a message without heading information to a file or folder. Displays the file name and the size of the file when the operation is complete. Does not include message headers in the file.
xit	(**x**) Identical to the **exit** subcommand.
z [**+** | **-**]	Changes the current message group (group of 20 messages) and displays the headings of the messages in that group. If a + or no argument is given, then headings in the next group are shown. If a **-** argument is given, the headings in the previous group are shown.

Mail Editor Subcommands for the mail, Mail Command

By default, the Mail program treats lines beginning with the ~ (tilde) character as subcommands. The following list describes the subcommands used while in the mail editor. The editor recognizes subcommands only if you enter them at the beginning of a new line.

~?	Displays a summary of the **mail** subcommands.
~!*Command*	The command interpreter specified by **SHELL** will be invoked with two arguments: **-c** and *command*. The standard output of command will be inserted into the message.
~a	Inserts the value of the **sign** variable into the text of the message, followed by a newline character. Identical to **~i sign**.
~A	Inserts the value of the **Sign** variable into the text of the message, followed by a newline character. Identical to **~i Sign**.
~b *AddressList*	Adds names in *AddressList* to the list of addresses to receive blind copies of the message. The **~b** subcommand can only be used to add to, not change or delete, the contents of the `Bcc:` *List*.
~c *AddressList*	Adds names in *AddressList* to the list of people to receive copies of the message. The **~c** subcommand can only be used to add to, not change or delete, the contents of the `Cc:` *List*.
~d	Appends the contents of the **dead.letter** file to the end of the message.
~e	Starts the alternate editor using the message text as the input file. (This editor can be defined with the **set EDITOR=** statement in the Bourne shell.) When you exit that editor, you return to the mail editor, where you may add text, or send the message by exiting the Mail program.
~f [*MessageList*]	Includes a *MessageList* in the current message to forward the message to another user. This subcommand reads each message in the *MessageList* and appends it to the current end of the message, but does not indent the appended message. This subcommand is also used to append messages for reference whose margins are too wide to embed with the **~m** subcommand. This subcommand works only if you entered the mail editor from the mailbox prompt using either the **mail**, **reply**, or **Reply** subcommand.
~F [*MessageList*]	Equivalent of the **~f**, except that all headers will be included in the message, regardless of previous **discard**, **ignore**, and **retain** commands.
~h	Enables you to add or change information in all of the heading fields. The system displays each of the four heading fields, one at a time. You can view the contents of each field and delete or add information to that field. Press the Enter key to save any changes to the field and to display the next field and its contents.

~i *string*	Inserts the value of the named variable, followed by a newline character, into the text of the message. If the string is unset or null, the message will not be changed.
~m [*MessageList*]	Includes a *MessageList* in the current message for reference purposes. This subcommand reads each message in the *MessageList* and appends it to the current end of the message. The included message is indented one tab character from the usual left margin of the message. This subcommand works only if you entered the mail editor from the mailbox prompt using either the **mail**, **reply**, or **Reply** subcommand.
~M [*MessageList*]	Equivalent of the **~m**, except that all headers will be included in the message, regardless of previous **discard**, **ignore**, and **retain** commands.
~p	Displays the entire message, including header information.
~q	Quits the editor without sending the message. Saves the message in the **dead.letter** file in your home directory, unless the **nosave** option is set. The previous contents of the **dead.letter** file are replaced with the partially completed message. **Note:** You can also quit the editor by using the Interrupt (Ctrl+C) key sequence twice.
~r *File*	Reads the contents of a file into the current message.
~s *String*	Changes the subject field to the phrase specified in *String*. You cannot append to the subject field with this subcommand.
~t *AddressList*	Adds the addresses in *AddressList* to the `To:` field of the message. The **~t** subcommand can only be used to add to, not change or delete, the contents of the `To:` *List*.
~v	Starts the visual editor using the message text as the input file. This editor can be defined with the **set VISUAL=** statement in the Bourne shell.) When you exit that editor, you return to the mail editor where you may add text to the message, or send the message by exiting the Mail program.
~w *File*	Writes the message to the named file.
~x	Exits as with **~q**, except the message will not be saved in the **dead.letter** file.
~: *Subcommand*	Executes the subcommand specified by *Subcommand* and returns to the mail editor.
~| *Command*	Pipes the message through the command *Command* as a filter. If *Command* gives no output or terminates unusually, it retains the original text of the message. Otherwise, the output of *Command* replaces the current message. The **fmt** command is often used as *Command* to format the message.
~< *file*	Reads the contents of a file into the current message.
~<!Command	Allows you to run a shell command. The shell runs with the **-c** flag and the **Command** specified. The standard output of **Command** is inserted into the message.
~~	Allows you to use the ~ (tilde) character in a message without it being interpreted as a command prefix. The ~~ key sequence results in only one ~ character being sent in the message.

Files

$HOME/.mailrc	Contains the **mail** subcommands to customize the Mail program for a specific user.
$HOME/mbox	Contains your personal mailbox.
/usr/share/lib/Mail.rc	Contains the file with mail subcommands to change the Mail program for all users on the system.
/var/spool/mail/*	Contains system mailboxes for all users.
/usr/bin/mail	Contains the **mail** command.
/usr/bin/Mail	Contains the **Mail** command.
/usr/bin/mailx	Contains the **mailx** command.

Related Information

The **bellmail** command, **fmt** command, **pg** command, **sendmail** command.

The **.mailrc** file format.

Mail Overview, Creating and Sending Mail, Receiving and Handling Mail in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

Reading Mail, Replying to Mail, Displaying the Contents of Your Mailbox, Editing a Message, Folders in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

Customizing the Mail Program, Starting the Mail Editor, Displaying a Message While in the Mail Editor, Changing or Adding to the Heading Fields of a Message in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

Checking for Misspelling in the Mail Editor, Reformatting a Message in the Mail Editor, Changing Text Editors Used for Entering Messages in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

Mailbox Subcommands for the mail Command .

---

mailq Command

Purpose

Prints the contents of the mail queue.

Syntax

/usr/sbin/mailq [**-v**]

Description

The **mailq** and **MAILQ** commands print a list of messages that are in the mail queue. The first line printed for each message shows:

- The internal identifier used on this host for the message with a possible status character
- The size of the message in bytes
- The date and time the message is accepted into the queue
- the envelope sender of the message

The second line shows the error message that caused the message to be retained in the in the queue, it is not displayed if the message is being displayed for the first time. The status characters are either:

***** Indicates the job is being processed

X Indicates that the load is too high to process the job

- Indicates that the job is too young to process

The following lines show message recipients, one per line.

The **mailq** command is the same as the **sendmail -bp** command.

Specify the **-v** flag to display message priority.

Flags

-v Prints verbose information. This adds the priority of the message and a single character indicator (+ or blank) indicating wheather a warning message has been sent on the first line of the message. Additionally, extra lines may be intermixed with the recipents indicating the *controlling user* information; this shows who owns any program that are executed on behalf of this message and the name of the alias this command expanded from, if any.

Exit Status

The command returns the following exit values:

0 Exits successfully.

>0 An error occurred.

Examples

The **mailq** command prints two types of lists:

- The **mailq** command lists the mail queue as shown in the following example:

```
     Mail Queue (1 request)
---QID---- --Size-- -----Q-Time----- ------Sender/Recipient-----
AA02508        3    Thu Dec 17 10:01            root
            (User unknown)
                                    bad_user
```

- The **mailq -v** command lists the mail queue as follows:

```
     Mail Queue (1 request)
---QID---- --Size-- -Priority- ---Q-Time--- --Sender/Recipient--
AA02508        3      1005     Dec 17 10:01        root
            (User unknown)
                                          bad_user
```

The fields have the following meanings:

QID	Contains the message queue ID of the message.
Size	Contains the number of bytes in the body of the message (heading information not included).
Priority	Contains the priority of the message, based primarily on the size of the message.
Q-Time	Contains the time the message entered the queue.
Sender/Recipient	Contains the user ID of the sender and the recipient of the message. A message on the line between the sender and the recipient indicates the status of the message.

Files

/usr/sbin/mailq	Contains the **mailq** command.
/var/spool/mqueue directory	Contains the log file and temporary files associated with the messages in the mail queue.

Related Information

The **sendmail** command.

Managing the Mail Queue Files and Directories in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

mailstats Command

Purpose

Displays statistics about mail traffic.

Syntax

mailstats [**-o**][**-p**] [**-C** *CfFile*] [**-f** *StatFile*]

Description

The **mailstats** command displays the current mail statistics. The time at which the statistics started displays reads the information in the format specified by **ctime**. The statistics for each mailer are displayed on a single line, with the following fields:

`M`	Contains the mailer number.
`msgsfr`	Contains the number of messages received by the local machine from the indicated mailer.
`bytes_from`	Contains the number of Kbytes in the messages received by the local machine from the indicated mailer.
`msgsto`	Contains the number of messages sent from the local machine using the indicated mailer.
`bytes_to`	Contains the number of bytes in the messages sent from the local machine using the indicated mailer.
`msgsrej`	Contains the number of messages rejected.
`msgsdis`	Contains the number of messages discarded.
`Mailer`	Contains the name of mailer.

After the statistics are displayed, a line totaling the value of all of the mailers displays, preceeded with a **T**. This information is separated from the statistics by a line containing only **=** (equal characters). Another line preceeded with a **C** lists the number of connections.

Flags

-C *cfFile*	Specifies use of the *cfFile* instead of the default **sendmailcf** file.
-f *StatFile*	Specifies use of the *StatFile* instead of the statistics file specified in the **sendmail.cf** file.
-o	Specifies that the name of the mailer does not display in the output.
-p	Outputs information in program readable mode and clears the statistics.

Exit Status

0	Successful completion.
>0	An error occurred.

Files

/etc/mail/statistics	Default **sendmail** statistics file.
/etc/mail/sendmail.cf	Default **sendmail** configuration file.

Related Information

The **mailq** command and **sendmail** command.

Managing the Mail Queue Files and Directories in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How To Display Mailer Information, How to Log the Mailer Statistics in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

make Command

Purpose

Maintains, updates, and regenerates groups of programs.

Syntax

make [**-D***Variable*] [**-d** *Option*]] [**-e**] [**-i**] [**-k**] [**-n**] [**-p**] [**-q**] [**-r**] [**-S**] [**-s**] [**-t**] [**-f** *MakeFile ...*] [*Target ...*]

Description

The **make** command assists you in maintaining a set of programs. Input to the **make** command is a list of file dependency specifications.

There are four types of lines in a makefile: file dependency specifications, shell commands, variable assignments, and comments. In general, lines can be continued from one line to the next by ending them with a \ (backslash). The trailing newline character and initial white space on the following line are compressed into a single space.

File Dependency Specifications

Dependency lines consist of one or more targets, an operator, and zero or more prerequisites (sources). This creates a relationship where the targets depend on the prerequisites and are usually created from them. The exact relationship between the target and the prerequisite is determined by the operator that separates them. The operators are as follows:

: A target is considered out-of-date if its modification time is less than that of any of its prerequisites. Prerequisites for a target accumulate over dependency lines when this operator is used. The target is removed if the **make** command is interrupted, unless the target has the **.PRECIOUS** attribute.

:: If no prerequisites are specified, the target is always recreated. Otherwise, a target is considered out-of-date if any of its prerequisites were modified more recently than the target. Prerequisites for a target do not accumulate over dependency lines when this operator is used. The target is not removed if the **make** command is interrupted.

File dependency specifications have two types of rules, inference and target. Inference rules specify how a target is to be made up-to-date. These rules have one target with no / (slash) and a minimum of one . (period). Target rules specify how to build the target. These rules can have more than one target.

Makefile Execution

The **make** command executes the commands in the makefile line by line. As **make** executes each command, it writes the command to standard output (unless otherwise directed, for example, using the **-s** flag). A makefile must have a Tab in front of the commands on each line.

When a command is executed through the **make** command, it uses **make**'s execution environment. This includes any macros from the command line to the **make** command and any environment variables specified in the **MAKEFLAGS** variable. The **make** command's environment variables overwrite any variables of the same name in the existing environment.

Note: When the **make** command encounters a line beginning with the word **include** followed by another word that is the name of a makefile (for example, **include depend**), the **make** command attempts to open that file and process its contents as if the contents were displayed where the include line occurs. This behavior occurs only if the first noncomment line of the first makefile read by the **make** command is not the **.POSIX** target; otherwise, a syntax error occurs.

Comments: Comments begin with a # character, anywhere but in a shell command line, and continue to the end of the line.

Environment: The **make** command uses the **MAKEFLAGS** environment variable, if it exists.

Target Rules

Target rules have the following format:

```
target[target...] : [prerequisite...] [;command]
<Tab>command
```

Multiple targets and prerequisites are separated by spaces. Any text that follows the ; (semicolon) and all of the subsequent lines that begin with a Tab character are considered commands to be used to update the target. A new target entry is started when a new line does not begin with a Tab or # character.

Note: The list of prerequisites can be empty.

Special Targets

Special targets cannot be included with other targets; that is, they must be the only target specified. These targets control the operation of the **make** command. These targets are:

.DEFAULT This is used as the rule for any target (that was used only as a prerequisite) that the **make** command cannot figure out any other way to create. Only the shell script is used. The **<** (left angle bracket) variable of a target that inherits **.DEFAULT**'s commands is set to the target's own name.

.IGNORE Prerequisites of this target are targets themselves; this causes errors from commands associated with them to be ignored. If no prerequisites are specified, this is the equivalent of specifying the **-i** flag.

.**POSIX** Causes the **make** command to use a different default rules file. The file, **/usr/ccs/lib/posix.mk**, provides the default rules as specified in the POSIX standard.

.PRECIOUS Prerequisites of this target are targets themselves. **.PRECIOUS** prevents the target from being removed. If no prerequisites are specified, the **.PRECIOUS** attribute is applied to every target in the file. Usually, when **make** is interrupted (for example, with **SIGHUP**, **SIGTERM**, **SIGINT**, or **SIGQUIT**), it removes any partially made targets. If **make** was invoked with the **-n**, **-p**, or **-q** flags, the target is considered to have the **.PRECIOUS** attribute.

.SILENT Prerequisites of the target are targets themselves. This causes commands associated with the target to not be written to standard output before they are executed. If no prerequisites are specified, the **.SILENT** attribute is applied to every command in the file.

.SUFFIXES Use this name to add more suffixes to the list of file suffixes that **make** recognizes. Prerequisites of the target are appended to the list of known suffixes. If no suffixes are specified, any previously specified suffixes are deleted. These suffixes are used by the inference rules. To change the order of suffixes, you need to specify an empty **.SUFFIXES** entry and then a new list of **.SUFFIXES** entries. A makefile must not associate commands with **.SUFFIXES**.

Inference Rules

The **make** command has a default set of inference rules, which you can supplement or overwrite with additional inference rules definitions in the makefile. The default rules are stored in the external file, **/usr/ccs/lib/aix.mk**. You can substitute your own rules file by setting the **MAKERULES** variable to your own file name from the command line. The following line shows how to change the rules file from the command line:

```
make MAKERULES=/pathname/filename
```

Inference rules consist of target suffixes and commands. From the suffixes, the **make** command determines the prerequisites, and from both the suffixes and their prerequisites, the **make** command determines how to make a target up-to-date. Inference rules have the following format:

```
rule:
<Tab>command
...
```

where `rule` has one of the following forms:

.s1	A single-suffix inference rule that describes how to build a target that is appended with one of the single suffixes.
.s1.s2	A double-suffix inference rule that describes how to build a target that is appended with *.s2* with a prerequisite that is appended with *.s1*.

The *.s1* and *.s2* suffixes are defined as prerequisites of the special target, **.SUFFIXES**. The suffixes *.s1* and *.s2* must be known suffixes at the time the inference rule is displayed in the makefile. The inference rules use the suffixes in the order in which they are specified in **.SUFFIXES**. A new inference rule is started when a new line does not begin with a <Tab> or # character.

If `rule` is empty, for example:

```
rule: ;
```

execution has no effect, and the **make** command recognizes that the suffix exists, but takes no actions when targets are out-of-date.

A ~ (tilde) in the preceding rules refers to an SCCS file. Therefore, the rule, **.c~.o**, would transform an SCCS C language prerequisite file into an object file (**.o**). Because the **s.** of the SCCS file is a prefix, it is incompatible with the **make** command's suffix view. The ~ (tilde) is a way of changing any file reference into an SCCS file reference.

Libraries

A target or prerequisite can also be a member of an archive library and is treated as such if there are parentheses in the name. For example, *library(name)* indicates that *name* is a member of the archive library *library*. To update a member of a library from a particular file, you can use the format *.s1***.a**, where a file with the *.s1* suffix is used to update a member of the archive library. The **.a** refers to an archive library.

Using Macros

In makefiles, macro definitions are defined in the format:

```
variable=value
```

Macros can be displayed throughout the makefile, as follows:

- If a macro is displayed in a target line, it is evaluated when the target line is read.
- If a macro is displayed in a command line, it is evaluated when the command is executed.
- If a macro is displayed in a macro definition line, it is evaluated when the new macro is displayed in a rule or command.

If a macro has no definition, it defaults to **NULL**. A new macro definition overwrites an existing macro of the same name. Macros assignments can come from the following, in the listed order:

1. Default inference rules
2. Contents of the environment
3. Makefiles
4. Command lines.

> **Note:** The **-e** flag causes environment variables to override those defined in the makefile.

The **SHELL** macro is special. It is set by the **make** command to the path name of the **shell** command interpreter (**/usr/bin/sh**). However, if it is redefined in the makefile or on the command line, this default setting is overridden.

Note: The **SHELL** macro does not affect, and is not affected by, the **SHELL** environment variable.

Shell Commands

Each target can have associated with it a series of shell commands, usually used to create the target. Each of the commands in this script must be preceded by a Tab. While any target can be displayed on a dependency line, only one of these dependencies can be followed by a creation script, unless the **::** operator is used.

If the first, or first two characters, of the command line are one or all of @ (at sign), - (hyphen), and + (plus sign), the command is treated specially, as follows:

@ Causes the command not to be echoed before it is executed.
- Causes any nonzero exit status of the command line to be ignored.
+ Causes a command line to be executed, even though the options **-n**, **-q**, or **-t** are specified.

A command that has no metacharacters is directly executed by the **make** command. For example, the **make** command consigns the first command in the following example to the shell because it contains the > (greater than sign) shell metacharacter. The second command in the following example does not contain any shell metacharacters, so the **make** command executes it directly:

```
target: dependency
        cat dependency > target
        chmod a+x target
```

Bypassing the shell saves time, but it can cause problems. For example, attempting to execute a C shell script from within a makefile by setting the **SHELL** macro to `/bin/csh` will not work unless the command line also contains at least one shell metacharacter.

```
SHELL=/bin/csh

target: dependency
        my_csh_script
```

This makefile fails because the **make** command attempts to run `my_csh_script` instead of consigning it to the C shell.

Variable Assignments

Variables in the **make** command are much like variables in the shell and consist of all uppercase letters. The **=** operator assigns values to variables. Any previous variable is then overridden.

Any white space before the assigned value is removed; if the value is being appended, a single space is inserted between the previous contents of the variable and the appended value.

Variables are expended by surrounding the variable name with either { } (braces) or ( ) (parentheses) and preceding it with a $ (dollar sign). If the variable name contains only a single letter, the surrounding braces or parentheses are not required. This shorter form is not recommended.

Variable substitution occurs at two distinct times, depending on where the variable is being used. Variables in dependency lines are expanded as the line is read. Variables in shell commands are expanded when the **shell** command is executed.

The four classes of variables (in order of increasing precedence) are:

Environment	Variables defined as part of the **make** command's environment.
Global	Variables defined in the makefile or in included makefiles.
Command line	Variables defined as part of the command line.

Local — Variables defined specific to a certain target. The local variables are as follows:

$< Represents either the full name of a prerequisite that made a target out-of-date (inference rule), or the full name of a target (**.DEFAULT** rule).

$* Represents the file name section of a prerequisite that made a target out-of-date (in an inference rule) without a suffix.

$@ Represents the full target name of the current target or the archive file name part of the library archive target.

$% Represents a library member in a target rule if the target is a member of the archive library.

You can also use these local variables appended with **D** or **F**:

D Indicates that the local variable applies to the directory part of the name. This is the path name prefix without a trailing / (slash). For current directories, **D** is a . (period).

F Indicates that the local variable applies to the file name part of the name.

In addition, the **make** command sets or knows about the following variables:

$** A single **$ (dollar sign); that is, **$$** expands to a single dollar sign.

LANG Determines the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC**_) do not specify a locale.

LC_ALL Determines the locale to be used to override any values for locale categories specified by the setting of **LANG** or any other **LC**_ environment variable.

LC_CTYPE Determines the locale for the interpretation of sequences of bytes of text data as characters; for example, single- versus multibyte characters in arguments.

LC_MESSAGES Determines the language in which messages should be written.

MAKEFLAGS The environment variable, **MAKEFLAGS**, can contain anything that can be specified on **make**'s command line. Anything specified on **make**'s command line is appended to the **MAKEFLAGS** variable, which is then entered into the environment for all programs that **make** executes. Note that the operation of the **-f** and **-p** flags in the **MAKEFLAGS** variable is undefined. Command line flags take precedence over the **-f** and **-p** flags in this variable.

VPATH Allows you to specify a list of directories to search for prerequisites. The list of directories works like the **PATH** variable in the **SHELL**. The **VPATH** variable can specify multiple directories separated by colons. For example:

```
VPATH=src:/usr/local/src
```

This tells the **make** command to search for the following directories in the order given:

- The current directory (this happens even without **VPATH**)
- `src` (a subdirectory in the current directory)
- `/usr/local/src`.

Flags

-D*Variable* Sets the value of *Variable* to 1.

-d*Option* Displays detailed information about the files and times that **make** examines (debug mode). The **-d** flag without any options or with the *A* option displays all the debug information available. Individually selectable debug options follow:

A	Displays all possible debug information.
a	Displays debug information about archive searching and caching.
d	Displays debug information about directory searching.
g1	Displays debug information about input graph before making anything.
g2	Displays debug information about input graph after making everything, or before exiting on an error.
m	Displays debug information about making targets, including modification dates.
s	Displays debug information about suffix searching.
v	Displays debug information about variable assignments.

-e Specifies that environmental variables override macro assignments within makefiles.

-f *MakeFile* Specifies a makefile to read instead of the default makefile. If *MakeFile* is - (hyphen), standard input is read. Multiple makefiles can be specified and are read in the order specified.

-i Ignores nonzero exit of **shell** commands in the makefile. Equivalent to specifying - (hyphen) before each command line in the makefile.

-k Continues processing after errors are encountered, but only on those targets that do not depend on the target whose creation caused the error.

-n Displays commands, but does not run them. However, lines beginning with a + (plus sign) are executed.

-p Displays the complete set of macro definitions and target descriptions before performing any commands.

-q Returns a zero status code if the target file is up-to-date; returns a one status code if the target file is not up-to-date. However, a command line with the + (plus sign) prefix will be executed.

-r Does not use the default rules.

-S Terminates the **make** command if an error occurs. This is the default and the opposite of **-k** flag.

-s Does not display commands on the screen as they are performed.

-t Creates a target or updates its modification time to make it seem up-to-date. Executes command lines beginning with a + (plus sign).

Target Specifies a target name of the form *Target* or sets the value of variables.

Exit Status

When the **-q** flag is specified, this command returns the following exit values:

0 Successful completion.
1 The target was not up-to-date.
>1 An error occurred.

Otherwise, this command returns the following exit values:

0 Successful completion.
>1 An error occurred.

Examples

1. To make the first target found in the makefile, type:

   ```
   make
   ```

2. To display, but not run, the commands that the **make** command would use to make a file:

```
make  -n search.o
```

Doing this will verify that a new description file is correct before using it.

3. To create a makefile that says that **pgm** depends on two files, **a.o** and **b.o**, and that they, in turn, depend on their corresponding prerequisite files (**a.c** and **b.c**) and a common file, **incl.h**, type:

```
pgm: a.o b.o
          c89 a.o b.o -o pgm
a.o: incl.h a.c
          c89 -c a.c
b.o: incl.h b.c
          c89 -c b.c
```

4. To make optimized **.o** files from **.c** files, type:

```
.c.o:
          c89 -c -o $*.c
or:
.c.o:
          c89 -c -o $<
```

5. To view the contents of the built-in rules, type:

```
make  -p  -f /dev/null 2>/dev/null
```

Files

makefile	Contains a list of dependencies.
Makefile	Contains a list of dependencies.
s.makefile	Contains a list of dependencies. It is an SCCS file.
s.Makefile	Contains a list of dependencies. It is an SCCS file.
/usr/ccs/lib/posix.mk	Contains default POSIX rules for the **make** command.
/usr/ccs/lib/aix.mk	Contains default rules for the **make** command.

Related Information

The **sh** command.

The make Command Overview in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

The Commands Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

makedbm Command

Purpose

Makes a Network Information Services (NIS) database map.

Syntax

To Create an NIS Map

/usr/sbin/makedbm [**-b**] [**-i** NIS*InputFile*] [**-o** NIS*OutputFile*] [**-d** NIS*DomainName*] [**-m** NIS*MasterName*] *InputFile OutputFile*

To Create a Non-dbm Formatted NIS Map

/usr/sbin/makedbm [**-u** *dbmFileName*]

Description

The **makedbm** command makes an NIS map. It does this by converting the file named in the *InputFile* parameter into two output files: *OutputFile.pag* and *OutputFile.dir*. Each line in each input file is converted into a single Data Base Manager (DBM) record.

The **makedbm** command is most often invoked from the **/var/yp/Makefile** file to generate NIS maps. All characters leading up to the first space or tab in each line of the **/var/yp/Makefile** file form the key. The rest of the line contains value data. If a line ends with a \ (backslash), data for that record is continued on the next line. NIS clients must interpret the # (pound sign) symbol since the **makedbm** command does not treat it as a comment character. If the *InputFile* parameter is a - (minus sign), the **makedbm** command reads standard input instead.

This command generates a special entry in the output map by using the **YP_LAST_MODIFIED** key, which is the date that the file specified by the *InputFile* parameter was created (or the current time, if the *InputFile* parameter is a - (minus sign)).

Flags

-b Propagates a map to all servers using the **named** name server.
-i Creates a special entry with the **YP_INPUT_FILE** key.
-o Creates a special entry with the **YP_OUTPUT_FILE** key.
-d Creates a special entry with the **YP_DOMAIN_NAME** key.
-m Creates a special entry with the **YP_MASTER_NAME** key.
-u Undoes a DBM file. That is, prints out a DBM file one entry per line, with a single space separating keys from values.

Files

/var/yp/Makefile	Contains rules for making NIS maps.

Related Information

The **ypinit** command, **yppush** command.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How to Create a Non-Standard NIS Maps in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

List of NDBM and DBM Programming References in *AIX 5L Version 5.2 Communications Programming Concepts*.

NIS Reference in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

---

makedepend Command

Purpose

Create dependencies in makefiles**.**

Syntax

makedepend [**-D***Name*=*Def*] [**-D***Name*] [**-I***IncludeDir*] [**-Y***IncludeDir*] [**-a**] [**-f***MakeFile*] [**-o***ObjSuffix*] [**-p***ObjPrefix*] [**-s***String*] [**-w***Width*] [**-v**] [**-m**] [**—***Options***—**] *SourceFile* ...

Description

The **makedepend** command reads each *SourceFile* in sequence and parses it like a C-preprocessor. It processes all #include, #define, #undef, #ifdef, #ifndef, #endif, #if, and #else directives to determine which #include directives need to be used in a compilation. Any #include directives can reference files having other #include directives, and parsing occurs in these files as well.

Every file that a *SourceFile* includes, directly or indirectly, is what **makedepend** calls a ″dependency.″ These dependencies are then written to a makefile in such a way that the **make** command can determine which object files must be recompiled when a dependency has changed.

By default, **makedepend** places its output in the file named **makefile** if it exists, otherwise **Makefile**. An alternate makefile may be specified with the **-f** flag. **makedepend** first searches the available makefile for the line:

```
# DO NOT DELETE THIS LINE - make depend depends on it.
```

or one provided with the **-s** flag, as a delimiter for the dependency output. If it finds the line, it deletes everything following the line to the end of the makefile and puts the output after the line. If **makedepend** does not find the line, it appends the delimited string to the end of the makefile and places the output immediately after the string.

For each *SourceFile* appearing on the command line, **makedepend** puts lines in the makefile in the following form.

```
SourceFile.o: dfile ...
```

Where `SourceFile.o` is the name from the command line with its suffix replaced with `.o`, and `dfile` is a dependency discovered in an #include directive while parsing the *SourceFile* or one of the files it included.

The algorithm used in this command assumes that all files compiled by a single makefile will be compiled with roughly the same **-I** and **-D** flags, and that most files in a single directory will include largely the same files.

Given these assumptions, **makedepend** expects to be called once for each makefile, with all source files that are maintained by the make file appearing on the command line. It parses each source and include file only once, maintaining an internal symbol table for each. As a result, the first file on the command line takes an amount of time proportional to the amount of time that a normal C preprocessor takes. On subsequent files, if it encounters an include file that it has already parsed, it does not parse again.

For example, imagine you are compiling two files, **file1.c** and **file2.c**, each includes the header file **header.h**. The **header.h** file includes the files **def1.h** and **def2.h**. When you run the command:

```
makedepend file1.c file2.c
```

then **makedepend** will first parse **file1.c** and consequently, **header.h** and then **def1.h** and **def2.h**. It then decides that the dependencies for this first file are:

```
file1.o: header.h def1.h def2.h
```

But when the program parses the second file, **file2.c** and discovers that it, too, includes **header.h**, it does not parse the file, but simply adds **header.h**, **def1.h** and **def2.h** to the list of dependencies for **file2.o**.

> **Note:** If you do not have the source for cpp (the Berkeley C preprocessor), then **makedepend** will compile in such a way that all #if directives will evaluate to False, regardless of their actual value.

This may cause the wrong #include directives to be evaluated. In these cases, it is recommended that you write a new parser for #if expressions. The need for a new parser should be clear from the following example:

Imagine you are parsing two files **file1.c** and **file2.c**, each includes the file **def.h**. The list of files that **def.h** includes might be very different when **def.h** is included by **file1.c** than when it is included by **file2.c.** But once **makedepend** arrives at a list of dependencies for a file, it is cast in concrete.

Flags

Note: The **makedepend** command ignores flags it does not understand. Flag usage is similar to that of the **cc** command.

-D*Name*=*Def* or **-D***Name*	Places a definition for the *Name* variable in the **makedepend** command's symbol table. Without the *=Def* specifier, the symbol is defined as 1.
-I*IncludeDir*	Prepends the *IncludeDir* variable to the list of directories searched by the **makedepend** command when it encounters an #include directive. By default, the **makedepend** command searches only the **/usr/include** directory**.**
-Y*IncludeDir*	Replaces all of the standard include directories with a single specified include directory, you can omit *IncludeDir* to prevent searching the standard include directories.
-a	Appends the dependencies to the end of the file instead of replacing them.
-f*MakeFile*	Enables you to specify an alternate makefile in which to place command output.
-o*ObjSuffix*	Specifies an object suffix. For example, some systems may have object files whose suffix is something other than **.o**. This flag allows you to specify another suffix, such as ″.b″ with **-o.b** or ″:obj″ with **-o.obj** and so forth.
-p*ObjPrefix*	Prepends the object file prefix to the name of the object file. This flag is used to designate a different directory for the object file. The default is the empty string.
-s*String*	Specifies the starting string delimiter. This flag permits you to specify a different string for **makedepend** to search for in the makefile.
-w*Width*	Changes the maximum line width of output lines. The default maximum is 78 characters.
-v	Causes **makedepend** to display a list of files included by each input file on standard input.
-m	Causes **makedepend** to display a warning if any input file includes another file more than once. In previous version of **makedepend** this was the default behavior. This flag is provided for backward compatibility and to aid in debugging problems related to multiple inclusion.
—*Options***—**	Ignores any unrecognized argument contained within a beginning and ending double hyphen. When **makedepend** encounters a double hyphen (—) in the argument list, any unrecognized argument following it is silently ignored; a second double hyphen terminates this treatment. The double hyphens enable **makedepend** to safely ignore esoteric compiler arguments that might normally be found in a CFLAGS **make** command macro (see the Examples section). All flags that **makedepend** recognizes and that appear between the pair of double hyphens are processed normally.

Examples

Normally, **makedepend** will be used in a makefile target so that typing `makedepend` updates the dependencies for the makefile.

```
SRCS=file1.c file2.c ...
CFLAGS=-O -DHACK -I../foobar -xyz
depend:
      makedepend -- $(CFLAGS) -- $(SRCS)
```

Related Information

The **cc** command, **make** command.

---

makedev Command

Purpose

Creates binary description files suitable for reading by the **troff** command and its postprocessors.

Syntax

makedev DESC | *FontFile ...*

Description

The **makedev** command creates binary files suitable for reading by the **troff** command and its postprocessors. When the **DESC** file is specified, the **makedev** command creates a **DESC.out** file and a set of font description files using the information contained in the **DESC** file. When a font file is specified, the **makedev** command creates the corresponding font description file.

Options

DESC Causes a **DESC.out** file to be created.

Parameters

FontFile Causes a *FontFile***.out** file to be created.

Examples

The following command:

```
makedev B
```

creates a **B.out** file, which contains the font tables for the Times-Bold fonts.

Related Information

The **troff** command.

The troff Font File Format.

---

makekey Command

Purpose

Generates an encryption key.

Syntax

makekey [*ASCIICharacters* ...]

Description

The **makekey** command generates an encryption key for use with programs that perform encryption. Its input and output are usually pipes.

The **makekey** command reads 10 characters from standard input and writes 13 characters to standard output. The first 8 of the 10 input characters can be any sequence of ASCII characters, as specified by the *ASCIICharacters* parameter. The last two input characters, called the salt, are chosen from the sets 0 through 9, a through z, A through Z, . (period), and / (slash). The salt characters are repeated as the first two characters of the output. The remaining 11 output characters are chosen from the same set as the salt and constitute the output key that you use as the encryption key parameter of programs that perform encryption.

Examples

Entering the following example at the command line generates an encryption key:

```
makekey
1234567890
```

The **makekey** command generates an encryption key based on the input string `1234567890` and displays the following encryption key below, with the `$_` (shell prompt) appearing immediately after the generated key and on the same line.

```
90y744T/NXw1U$_
```

Related Information

The **crypt, encrypt, or setkey** subroutine.

---

makemap Command

Purpose

Creates database maps for **sendmail**.

Syntax

makemap [**-C -N -c** *CacheSize* **-d -e -f -l -o -r -s -u -v**] *Maptype Mapname*

Description

The **makemap** command creates the database maps used by the keyed map lookups in the **sendmail** command. It reads input from the standard input and outputs them to the indicated *Mapname*.

Parameters

Maptype — Depending upon how it is compiled, this command handles up to three different database formats:

- **dbm** DBM format maps. This requires the **ndbm** library.
- **btree** B-tree format maps. This requires the new Berkeley DB library.
- **hash** Hash format maps. This requires the new Berkeley DB library.

> **Note:** In all cases, this command reads lines from the standard input, consisting of two words separated by white space. The first is the database key, the second is the value. The value may contain ''%n'' strings to indicate parameter substitution. Literal percents should be doubled (″%%″). Blank lines and lines beginning with a ″#″ are ignored.

Mapname — Name of the map.

Flags

-c *CacheSize* — Specifies to use the hash and B-tree cache size.

-C	Indicates to use the specified **sendmail** configuration file for looking up the **TrustedUser** option.
-d	Allows duplicate keys in the map. This is only allowed on B-Tree format maps. If two identical keys are read, they are both inserted into the map.
-e	Allows empty values, right hand side.
-f	Disables the function of folding all upper case letters in the key to lowercase. This is intended to mesh with the **-f** flag in the **K** line in the **sendmail.cf** file. The value is never case folded.
-l	Lists supported map types.
-N	Includes the Null byte that terminates strings in the map. This must match the **-N** flag in the sendmail.cf ″K″ line.
-o	Specifies to append to an existing file. This allows you to augment an existing file.
-r	Allows replacement of existing keys. Normally, the **makemap** command complains if you repeat a key and does not perform the insert.
-s	Iignores safety checks on maps being created. This includes checking for hard or symbolic links in world writeable directories.
-u	Dumps, unmaps, the contents of the database to standard output.
-v	Specifies that the command verbosely print its status.

Related Information

The sendmail command.

---

man Command

Purpose

Displays manual entries online.

Syntax

man [[[**-c**] [**-t**] [*Section*]] | [**-k** | **-f**]] [**-M***Path*] [**-r**] *Title ...*

Description

The **man** command provides reference information on topics, such as commands, subroutines, and files. The **man** command provides one-line descriptions of commands specified by name. The **man** command also provides information on all commands whose descriptions contain a set of user-specified keywords.

The **man** command formats a specified set of manual pages. If you specify a section for the *Section* parameter, the **man** command searches in that section of the manual pages for the title specified by the *Title* parameter. The value of the *Section* parameter can be either an Arabic number from 1 through 8 or a letter.

The Section letters are:

C	Specifies commands (including system management commands).
F	Specifies file-type manual pages.
L	Specifies library functions.
n	Specifies new.
l	Specifies local.
o	Specifies old.
p	Specifies public.

Note: The **n**, **l**, **o**, and **p** section specifiers are not valid for reading the hypertext information bases, which contain the operating system documentation.

The Section numbers are:

1 Indicates user commands and daemons.
2 Indicates system calls and kernel services.
3 Indicates subroutines.
4 Indicates special files, device drivers, and hardware.
5 Indicates configuration files.
6 Indicates games.
7 Indicates miscellaneous commands.
8 Indicates administrative commands and daemons.

Note: The operating system documentation in the hypertext information databases is grouped into three sections only: command manual pages (in section 1, equivalent to section C), subroutine manual pages (in section 3, equivalent to section L), and file manual pages (in section 4, equivalent to section F). When searching for hypertext information, specifying section 1, 6, 7, or 8 will default to the command manual pages, section 2 or 3 will default to the subroutine manual pages, and section 4 or 5 will default to the file manual pages.

If the *Section* parameter is omitted, the **man** command searches all sections of the manual.

The search path the **man** command uses is a list of directories separated by a : (colon) in which manual subdirectories can be found. The **MANPATH** environment variable value is used for the default path. The **MANPATH** environment variable is not valid when reading the hypertext information bases.

The **man** command displays the manual pages as follows:

1. The **man** command searches the **nroff** directories (**man?**) under the **/usr/share/man** directory.
2. The **man** command searches the formatted version directories (**cat?**) under the **/usr/share/man** directory. If the formatted version is available, and if it has a more recent modify time than the **nroff** command source, the **man** command displays the formatted version. Otherwise, the manual page is formatted with the **nroff** command and displayed. If the user has permission, the formatted manual page is deposited in the proper place, so that later invocations of the **man** command do not format the page again.

   Note: There is no **nroff** source for the supplied manual pages. However, you can put **nroff** source for manual pages into the **man** directories and the **man** command can locate and process the **nroff** source.
3. If the **man** command does not find a manual page in the **/usr/share/man/man** or /**usr/share/man/cat** directory, the **man** command reads from the hypertext information bases. The hypertext information bases reside in the **/usr/share/man/info** directory structure and contain the operating system documentation. When reading from the hypertext databases, the **man** command does not put any manual pages in the **/usr/share/man/cat** directory structure. The **man** command strips formatting information from the manual page, wraps lines so the lines fit on the display, and displays the manual page using the command described by the **PAGER** environment variable.

When accessing the HTML databases, **man** looks for the operating system library before it proceeds to other LPP libraries. Within these libraries, it processes information in the following order:

cmds	Commands Reference
libs	Subroutines, System Calls
files	Files Reference

If the standard output is a tty, the **man** command pipes its output using the **more** command with the **-s** and **-v** flags. The **-s** flag eliminates multiple blank lines and stops after each page on the screen. The **-v**

flag suppresses the display of nonprinting characters to the screen. To continue scrolling, press the space bar. To scroll an additional 11 lines when the output stops, press the Ctrl-D key sequence.

The **PAGER** environment variable can be set to whatever pager is desired. The default value is the **more** command. To change the default pager, enter:

```
PAGER=Somepager
export PAGER
```

For example, if there are customized manual pages which are formatted with reverse or fractional line feeds, the **PAGER** environment variable may be set to **/usr/bin/pg** so that the line feeds are not printed as control characters. This procedure is not necessary for the manual pages.

When the **man** command uses a hypertext database, it can retrieve several articles. For example, `man open` displays several articles. The use of **SIGINT** (Ctrl-C) exits the **man** command completely. On the other hand, `man open close` also displays several articles but the use of **SIGINT** (Ctrl-C) causes **man** to display the **close** command information instead of exiting. Using **SIGINT** (Ctrl-C) again exits the **man** command completely.

When specifying one of the Network Computing System library routines that contains a **$** (dollar sign) in its name, enter a \ (backslash) preceding the **$**.

Flags

-c Displays the manual information using the **cat** command.

-f Displays entries in the keyword database related only to the command name given as the final parameter. You can enter more than one command name, each separated by a space. Use this flag to search for command articles only. To use the **-f** flag, a root user must have previously entered `catman -w` to create the **/usr/share/man/whatis** file.

-k Displays each line in the keyword database that contains a string of characters matching the title given as the final parameter. You can enter more than one title, each separated by a space. To use the **-k** flag, a root user must have previously entered `catman -w` to create the **/usr/share/man/whatis** file.

-M*Path* Changes the standard location where the **man** command searches for manual information. The search path the **man** command uses is a list of directories separated by a : (colon) in which manual subdirectories can be found. The **MANPATH** environment variable value is used for the default path.
Note: The **-M** flag is not valid when the **man** command reads from the hypertext databases.

-r Searches remotely for the manual information. If for any reason (either the remote machine is not reachable, or there is a problem reading URL or the Java is not installed on the local machine) the remote search fails, then **man** performs a local search for the requested man page.
Note: The **DOCUMENT_SERVER_MACHINE_NAME** environment variable should be set to the name of the documentation search server machine the user wants to use. If the AIX Base Documentation is not supported for the host's locale, the **man** command searches for the documentation for an alternate locale. If the search is successful, the documentation page is displayed after conversion to the local host's locale. If the alternate locale is not installed on the local host, the **man** command fails to display the documentation page.

-t Formats the manual information using the **troff** command. This flag is ignored if the manual page is found in a hypertext information base.

Exit Status

This command returns the following exit values:

0 Successful completion.

>0 An error occurred.

Examples

1. To display information about the **grep** command, enter:

   ```
   man grep
   ```

2. To display information about the **rpc_$register** library routine, enter:

   ```
   man rpc_\$register
   ```

3. To display all entries in the **/usr/share/man/whatis** keyword database that contain the ″mkdir″ string, enter:

   ```
   man -k mkdir
   ```

   The output is equivalent to the **apropos** command. You receive output from the **-k** flag only when the **/usr/share/man/whatis** keyword database already exists.

4. To display all entries from the keyword database related to the **nroff** and **troff** commands, enter:

   ```
   man -f nroff troff
   ```

   The output is equivalent to the **whatis** command. You receive output from the **-f** flag only when the **/usr/share/man/whatis** keyword database already exists.

5. To display all **ftp** command related articles in the **/usr/share/man** or **/usr/share/man/local** path, enter:

   ```
   man -M/usr/share/man:/usr/share/man/local ftp
   ```

Files

/usr/bin/man	Contains the **man** command.
/usr/share/man	Standard manual directory structure.
/usr/share/man/cat?/*	Directory containing preformatted pages.
/usr/share/man/whatis	Contains the keyword database.
/usr/share/man/man?/*	Directory containing **nroff** format manual pages.

Related Information

The **apropos** command, **catman** command, **more** command, **whatis** command, **whereis** command.

---

managefonts Command

Purpose

Provides the user with a simple menu-based interface to update or change the set of installed font families on the system.

> **Note:** You must have root user authority to run the **managefonts** script. The **managefonts** script is contained in the **/usr/lib/ps/ditroff.fonts/managefonts** file.

Syntax

managefonts [*Option*]

Description

The **managefonts** command provides the user with a simple menu-based interface to update or change the set of installed font families on the system. If no command line arguments are provided, the menu-based interface is used. Command-line arguments can be used to provide the equivalent of the menu selections.

A set of font families is installed on the system at the time the TranScript Tools option of the Text Formatter Services Package is installed on the system. This default setup includes the standard 13 fonts

comprising the Times, Courier, and Helvetica font families. You can use the program called up by the **managefonts** command to erase the current configuration and replace it with a new one. There are several predefined packages of font families that can be installed this way:

Times Family Only	This is the most minimal configuration that allows the TranScript Tools option to run.
Standard13 Package	This package builds the Times, Courier, and Helvetica font families. This was the package installed on your system with TranScript.
Standard35 Package	This font family package includes the Standard13 package font families in addition to the following: Avant Garde, Bookman, New Century Schoolbook, and Palatino font families.
All Font Families	This package installs all the font families available for installation.

You can also use the **managefonts** command to add new font families one at a time. A menu of available fonts is displayed and users can select which font family they want to be built. The program prevents building of font families that are already installed.

The **managefonts** command includes help screens to assist the user in installing font families.

Notes:

1. Font families cannot be deleted directly. To delete font families, it is first necessary to install a package containing the minimal subset of families desired. After the package is installed, it is possible to add font families, one at a time, from the Individual Fonts Menu. For instance, if your current configuration is Times, Courier, and Helvetica, and you want only Times and Courier, you can use the **managefonts** program to install the Times Only Package.
2. There is no command-line syntax equivalent to the menu items in the **managefonts** program.

The command line arguments are acted upon in the order they are given, reading left to right. The following are the valid values for the *option* parameter and their meanings:

init0	Initialize for the installation of a font package.
clean	Remove all temporary files and previously installed fonts.
cleanall	Remove all the temporary files, the previously installed fonts, and the TranScript troff font files installed.
default	Install the Standard 13 fonts.
standard13	Install the Standard 13 fonts.
standard35	Install the Standard 35 fonts.
all	Install all possible fonts.
CourierFamily	Install the Courier Family.
HelveticaFamily	Install the Helvetica Family.
HelvNarrowFamily	Install the Helvetica Narrow Family.
AvantGardeFamily	Install the Avant Garde Family.
BookmanFamily	Install the Bookman Family.
GaramondFamily	Install the Garamond Family.
LubalinFamily	Install the Lubalin Family.
NewCenturyFamily	Install the New Century Family.
OptimaFamily	Install the Optima Family.
PalatinoFamily	Install the Palatino Family.
SouvenirFamily	Install the Souvenir Family.
ZapfFamily	Install the Zapf Family.
BaseFamily	Install the Base Family, such as Times Roman.

Examples

1. To install the standard 13 fonts:
   ```
   managefonts cleanall standard13
   ```
2. To install the standard 35 fonts:
   ```
   managefonts cleanall standard35
   ```
3. To install all the fonts:
   ```
   managefonts cleanall all
   ```
4. To install the Courier Family (the Times Roman or Base Family must have been previously installed):
   ```
   managefonts init0 CourierFamily clean
   ```

Related Information

The **enscript** command, **ps630** command, **psrev** command, **ps4014** command, **psdit** command, **psplot** command.

---

mant Command

Purpose

Typesets manual pages.

Syntax

mant [**-M** *Media*] [**-a**] [**-c**] [**-e**] [**-t**] [**-z**] [**-T** *Name*] [*troffFlags*] [*File* ... | **-**]

Description

The **mant** command uses the manual page macros (**man** macro package) to typeset manual pages. The *File* parameter specifies the files to be processed by the **mant** command. Files must be displayed after all flags. If no file name is specified, the **mant** command prints a list of its flags. If a **-** (minus sign) is specified for the *File* parameter, standard input is read.

The **mant** command has flags to specify preprocessing by the **tbl** command, **cw** command, or **eqn** command. Flags from the **troff** command can be specified with the *troffFlags* parameter.

If the input contains a **troff** command comment line consisting solely of the string `'\" x` (single quotation mark, backslash, double quotation mark, *x*), where `x` is any combination of the three letters **c**, **e**, and **t**, and where there is exactly one character space between the double quotation mark and `x`, then the input is processed through the appropriate combination of the **cw** command, **eqn** command, and **tbl** command, respectively, regardless of the command-line options.

> **Note:** Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output. Calling the **mant** command with one or more of the **-c** flag, **-e** flag, **-t** flag, and **-** (minus) flags together with the **-o***List* flag of the **troff** command, give a `broken pipe` message if the last page of the document is not specified by the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

The **mant** command, unlike the **troff** command, automatically pipes its output to a specific postprocessor, according to the following flags, environment variable, or default setting unless specifically requested not to do so:

-z	Indicates that no postprocessors are used.
-T*Name*	Prepares the output for the printing device specified by the *Name* variable.
TYPESETTER	Specifies a particular printing device for the system environment.
default	Sends to **ibm3816**.

Flags, other than the ones in the following list, are passed to the **troff** command or to the macro package, as appropriate. All flags must be displayed before the specified file names.

Flags

All flags must appear before the specified file names.

-a Calls the **-a** flag of the **troff** command.

-c Preprocesses the input files with the **cw** command.

-e Preprocesses the input files with the **eqn** command.

-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.

-t Preprocesses the input files with the **tbl** command.

-z Prepares the output without the postprocessor.

-T*Name* Prepares the output for the specified printing device. Possible *Name* variables are:

ibm3812
3812 Pageprinter II.

ibm3816
3816 Pageprinter.

hplj Hewlett-Packard LaserJet II.

ibm5587G
5587-G01 Kanji Printer multi-byte language support.

psc PostScript printer.

X100 AIXwindows display.

- Forces input to be read from standard input.

Related Information

The **cw** command, **eqn** command, **nroff** command, **tbl** command, **tc** command, **troff** command.

---

mark Command

Purpose

Creates, modifies, and displays message sequences.

Syntax

mark [**+***Folder*] [**-list**] [**-sequence** *Name* [*Messages*...] [**-add** | **-delete**] [**-zero** | **-nozero**] [**-public** | **-nopublic**]]

Description

The **mark** command creates, deletes, adds, and lists messages in a sequence. The **mark** command by default lists all of the sequences and their messages for the current folder. If you use the **-add** or **-delete** flag, you must also use the **-sequence** flag. When all messages are deleted from a sequence, the **mark** command removes the sequence name from the folder.

To create a new sequence, enter the **-sequence** flag with the name of the sequence you want to create. The **mark** command creates the sequence starting with the current message. By default, the **mark** command places the sequence in the current folder. If you specify a folder, that folder becomes the current folder.

Flags

-add Adds messages to a sequence. The **-add** flag is the default. If you do not specify a message, the **mark** command uses the current message.

Note: You can only use this flag with the **-sequence** flag.

-delete Deletes messages from a sequence. If you do not specify a message, the current message is deleted by default.

Note: You can only use this flag with the **-sequence** flag.

+*Folder* Specifies the folder to examine.

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

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

-list Displays the messages in a sequence. By default, the **-list** flag displays all the sequence names and messages defined for the current folder. To see a specific sequence, use the **-sequence** flag with the **-list** flag.

-nopublic Restricts a sequence to your usage. The **-nopublic** flag does not restrict the messages in the sequence, only the sequence itself. This option is the default if the folder is write-protected from other users.

-nozero Modifies the sequence by adding or deleting only the specified messages. This flag is the default.

-public Makes a sequence available to other users. The **-public** flag does not make protected messages available, only the sequence itself. This flag is the default if the folder is not write-protected from other users.

-sequence *Name* Specifies a sequence for the **-list**, **-add**, and **-delete** flags.

-zero Clears a sequence of all messages except the current message. When the **-delete** flag is also specified, the **-zero** flag places all of the messages from the folder into the sequence before deleting any messages.

Messages Specifies messages in a sequence. You can specify more than one message at a time. Messages are identified with following references:

Number
: Number of the message

all All the messages in a folder

cur or . (period)
: Current message (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

If the **-list** flag is used, the default for the *Messages* parameter is **all**. Otherwise, the default is the current message.

Profile Entries

The following entry is found in the *UserMHDirectory*/**context** file:

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

The following entry is found in the **$HOME/.mh_profile** file:

`Path:` Specifies the MH directory.

Examples

1. To see the list of all sequences defined for the current folder, enter:

   ```
   mark
   ```

   The system displays a message similar to the following:

   ```
   cur: 94
   test: 1-3 7 9
   ```

   In this example, message `94` is the current message number in the current folder. The message sequence called `test` includes message numbers `1`, `2`, `3`, `7`, and `9`.
2. To see the list of all the sequences defined for the `meetings` folder, enter:

   ```
   mark  +meetings
   ```

   The system displays a message similar to the following:

   ```
   cur: 5
   dates: 12 15 19
   ```
3. To create a new message sequence called `schedule` in the current folder, enter:

   ```
   mark  -sequence schedule
   ```

   The system displays the shell prompt to indicate that the `schedule` sequence was created. By default, the system adds the current message to the new sequence.
4. To delete message `10` from the `schedule` sequence, enter:

   ```
   mark  -sequence schedule 10  -delete
   ```

Files

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

Related Information

The **pick** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

mesg Command

Purpose

Permits or refuses write messages.

Syntax

mesg [**n** | **y**]

Description

The **mesg** command controls whether other users on the system can send messages to you with either the **write** command or the **talk** command. Called without arguments, the **mesg** command displays the current workstation message-permission setting.

The shell startup process permits messages by default. You can override this default action by including the line `mesg n` in your **$HOME/.profile** file. A user with root user authority can send write messages to any workstation, regardless of its message permission setting. Message permission has no effect on messages delivered through the electronic mail system (**sendmail**).

If you add `mesg y` to your **$HOME/.profile**, you will be able to receive messages from other users via the **write** command or the **talk** command.

If you add `mesg n` to your **$HOME/.profile**, you will not be able to receive messages from other users using the **write** command or the **talk** command.

Flags

n Allows only the root user the permission to send messages to your workstation. Use this form of the command to avoid having others clutter your display with incoming messages.

y Allows all workstations on the local network the permission to send messages to your workstation.

Exit Status

This command returns the following exit values:

0 Receiving messages is allowed.

1 Receiving messages is not allowed.

>1 An error occurred.

Examples

1. To allow only the root user the permission to send messages to your workstation, enter:

   ```
   mesg  n
   ```

2. To allow everyone the permission to send messages to your workstation, enter:

   ```
   mesg  y
   ```

3. To display what your current message-permission setting is, enter:

   ```
   mesg
   ```

   Information similar to the following is displayed:

   ```
   is  y
   ```

In the previous example, the current message-permission setting is `y` (allowing all users on the local network the permission to send messages to your workstation). If you change the message-permission setting to `n` (allowing only the root user the permission to send messages to your workstation), information similar to the following is displayed:

```
is  n
```

Files

/dev/tty*	Supports the controlling terminal interface.
$HOME/.profile	Controls startup processes and daemons.

Related Information

The **sendmail** command, **talk** command, **write** command.

Network Overview in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

mgmtsvr Command

Purpose

mgmtsvr – Displays or sets the management server entry for the Managed node.

Syntax

mgmtsvr [**–h**] [**–k**] [**–v** | **–V**] [**–N**] [**–n** *hostname*] [**host** | **–d**]

Description

The **mgmtsvr** command runs on Managed nodes to display or set the management server for the node.

Flags

–d Deletes the management server entry.

–h Writes the command's usage statement to standard output.

host Specifies the host name or IP address of the CSM management server that will be managing this node. If no host is specified, and the **–d** flag is not specified, the current setting will be displayed. In this case, the exit status can be examined to distinguish between nodes and the management server.

–k Exchanges RSCT public keys, host name, and node identifier between the node and the management server. Only use this flag if the host name, node identifier, or public key of either machines has changed. At the time this flag is used, you must trust your network to ensure that a rogue machine is not going to insert its public key in place of the public key of the real node.

–n *hostname*
Specifies the host name known to the CSM management server for this node. This flag is used internally by the **updatenode** command to ensure correct operation when the hostname value on the node resolves to a different name than the name known to the management server. The default is **/bin/hostname**.

–N Displays the value for the local node name and management server. This flag is valid only if a host name or IP address for the management server is not specified.

–v | –V
Writes the command's verbose messages to standard output.

Parameters

None.

Security

The command requires root access to the cluster management server.

Exit Status

0 This is a Managed node only and the management server value is set.

1 The combination of flags is not valid.

11 This is a Managed node only and the management server value is not set.

21 This node is its own management server.

22 This node is also a management server for a separate cluster.

23 This machine is a CSM management server and a node, but the node does not currently have a management server set for it.

31 This machine is a CSM management server only.

101 This machine is neither a CSM management server or node.

Environment Variables

DSH_PATH
Sets the path that is used on the remote nodes. If DSH_PATH is not set, the default path for the remote shell is used. For example, **DSH_PATH=$PATH** sets the path on the remote node to the same path that is used on the source node.

DSH_REMOTE_CMD
Specifies the path of the remote shell executable to use instead of the default.

DSH_REMOTE_OPTS
Includes the options specified in the remote command when the command is forwarded to the remote nodes.

DSH_FANOUT
Sets the maximum number of concurrent remote shell commands. This can also be set by the **–f** flag.

DSH_LIST
Specifies a file that contains definitions of the set of nodes that comprise the node list.

DSH_TIMEOUT
Specifies the time, in seconds, to wait for response from the remote shell. This can also be set with the **–t** flag.

WCOLL
Specifies a file that contains definitions of the set of nodes that comprise the working collective.

Standard Error

When the **dshbak** filter is used and standard error messages are generated, all error messages on standard error appear before all standard output messages. This is true with and without the **-c** flag.

Examples

To query the current CSM management server for this node, enter:

```
mgmtsvr
```

Location

/opt/csm/bin/mgmtsvr

Files

/var/log/csm/install.log

Related Information

- The **installnode** and **updatenode** commands.
- See the cluster security section in the *IBM CSM for AIX 5L: Administration Guide* for information on editing the RMC ACL file.

---

mhl Command

Purpose

Produces formatted listings of messages.

Syntax

mhl [**-form** *FormFile*] [**-folder** **+***Folder*] [**-moreproc** *Command* | **-nomoreproc** [**-bell** | **-nobell**] [**-clear** | **-noclear**]] [**-length** *Number*] [**-width** *Number*]

Description

The **mhl** command creates formatted lists of messages. The command is usually started through the `showproc:` profile entry or through the **-showproc** flag in other MH commands. When displaying messages, the **mhl** command uses the directions listed in the format file. If you specify more than one message, the **mhl** command provides a prompt before displaying each screen of messages.

If the **-nomoreproc** flag is specified, the **mhl** command prompts the user to press the Return key (the Ctrl-D key sequence is also acceptable) to see the next message. To stop the current message output and receive a prompt for the next message, press the Ctrl-D key sequence. Press the QUIT key sequence to stop the command output.

> **Note:** To use the **mhl** command, you must make the folder you wish to work with the current directory.

Flags

-bell	Produces a bell at the end of each page. When the **-nomoreproc** flag is specified or the `moreproc:` profile entry is defined, but empty, the **-bell** flag is the default.
-clear	Clears the screen after each page when the output device is a display. The **mhl** command uses the **$TERM** environment variable to determine the type of display. When the output device is not a display, the **-clear** flag inserts a form feed character at the end of each message. This flag affects the **mhl** command only if the `moreproc:` profile entry is defined and empty.
-folder **+***Folder*	Identifies the folder to be used for the **mhl.format** file's `MessageName:` entry. The default is the value of the **$mhfolder** environment variable.
-form *FormFile*	Specifies a file containing an alternate output format. The default format is described in the *UserMHDirectory***/mhl.format** file. If this file does not exist, the **mhl** command uses the system default format described in the **/etc/mh/mhl.format** file.
-help	Lists the command syntax, available switches (toggles), and version information.

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

-length *Number*	Sets the screen length for the output. The default is the value indicated by the **$TERM** environment variable. If that value is not appropriate, the default is 40 lines.
-moreproc *Command*	Uses the value of the *Command* variable instead of the value of the `moreproc:` entry specified in the **$HOME/.mh_profile** file.
-nobell	Suppresses the bell at the end of each page. This flag affects the **mhl** command only if the output device is a display, the **-nomoreproc** flag is used, or the `moreproc:` profile entry is defined and empty.
-noclear	Prevents clearing of the screen at the end of each page when the output device is a display. When the output device is not a display, the **-clear** flag does not insert a form-feed character at the end of each message. This flag is the default when the **-moreproc** flag is used or the `moreproc:` entry is defined and is empty.
-nomoreproc	Sets the `moreproc:` entry as an empty value.
-width *Number*	Sets the screen width for the output. The default is the value indicated by the **$TERM** environment variable. If that value is not appropriate, the default is 80 characters.

Profile Entries

The following entry is found in the *UserMHDirectory*/**.mh_profile** file:

`moreproc:` Specifies the interactive program for communicating with the user.

Examples

1. To list message `5` in the **inbox** folder, change the directory to **inbox**:

   ```
   cd /home/mickey/Mail/inbox
   ```

   Then enter:

   ```
   /usr/lib/mh/mhl 5
   ```

   A display similar to the following appears:

   ```
   --- Using template MHL.FORMAT ---
   Date:

   To:
   cc:

   From:
   Subject:

   Message Text
   ```

2. To display more than one message, enter:

   ```
   /usr/lib/mh/mhl 5 6 7
   ```

Files

$HOME/.mh_profile	Contains the MH user profile.
/etc/mh/mhl.format	Defines the default MH message template.
UserMHDirectory/**mhl.format**	Specifies a user's default message template. (If it exists, it overrides the default MH message template.)
/usr/lib/mh/mhl	Contains the **mhl** command.

Related Information

The **ap** command, **dp** command, **next** command, **prev** command, **show** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

mhmail Command

Purpose

Sends or receives mail.

Syntax

mhmail *User* ... [**-cc** *User* ...] [**-from** *User* ...] [**-subject** *"String"*] [**-body** *"String"*]

Description

The **mhmail** command composes, sends, and files messages. To file a message, enter the **mhmail** command without any flags. The default folder is **$HOME/inbox**.

If you specify one or more user addresses with the *User* parameter, the **mhmail** command accepts text from your terminal and composes a message. You can end the message text by pressing the Ctrl-D key sequence. The **mhmail** command sends a copy of the message to each specified address.

Flags

-body *"String"*	Sends a message with the specified string as the body. You must enclose the string in quotes. When you specify the **-body** flag, the **mhmail** command does not accept text from the terminal.
-cc *User...*	Sends a copy of the message to the specified users. The **mhmail** command puts the addresses in the `cc:` field.
-from *User...*	Places the specified user address in the `From:` field of the message.
-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-subject *"String"*	Places the specified text string in the `Subject:` field of the message.

Examples

1. To receive new mail and file it into the default mail folder, **$USER/Mail/inbox**, enter:

   ```
   mhmail
   ```

   The system displays a message similar to the following:

   ```
   Incorporating new mail into inbox...
   65+  04/08 jim@athena.a   Meeting        <<The meeting will
   66   04/08 jim@athena.a   Schedule       <<Schedule change
   ```

   In this example, two messages are filed in the `inbox` file. The subject of the first message is `Meeting`, and the first line starts with the words `The meeting will`. The subject of the second message is `Schedule`, and the first line starts with the words `Schedule change`.

2. To send a message regarding a schedule change to user `jamie` on system `venus`, enter:

   ```
   mhmail jamie@venus  -subject "Schedule Change"
   ```

The system waits for you to enter the text of the message. After completing the last line of the text, press the Enter key and then the Ctrl-D key sequence to send the message.

Files

/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/mhmail	Contains the **mhmail** command.

Related Information

The **inc** command, **post** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

mhpath Command

Purpose

Prints full path names of messages and folders.

Syntax

mhpath [**+***Folder*] [*Messages* [,*Messages*] ...]

Description

The **mhpath** command lists the path names of folders and messages. By default, the command lists the path name of the current folder.

Flags

+*Folder*	Specifies which folder path to list.
-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 for which you want to list path names. The *Messages* parameter can specify several messages, a range of messages, or a single message. Use the following references to specify messages.

Number
Number of the message. When specifying multiple messages, separate each message number with a comma. When specifying a range of messages, separate the upper and lower ends of the range with a hyphen.

Note: You cannot use the **new** variable when specifying a range.

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

all All the messages in a folder.

cur or . (period)
Current message.

first First message in a folder.

last Last message in a folder.

new Path name that the system will assign to the next message that is incorporated.

next Message following the current message.

prev Message immediately before the current message.

Profile Entries

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

`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies a user's MH directory.

Examples

1. To list the path name of the current folder, enter:

   ```
   mhpath
   ```

   The system responds with a message similar to the following:

   ```
   /home/tom/Mail/inbox
   ```

2. To list the path names for messages `2` through `4` in the **source** folder, enter:

   ```
   mhpath  +source 2-4
   ```

   The system responds with a message similar to the following:

   ```
   /home/tom/Mail/source/2
   /home/tom/Mail/source/3
   /home/tom/Mail/source/4
   ```

3. To list the path name the system will assign to the next message added to the current folder, enter:

   ```
   mhpath new
   ```

   The system responds with a message similar to the following:

   ```
   /home/tom/Mail/source/5
   ```

In this example, the next message will be message 5 in user tom's current folder, /home/tom/Mail/source.

Files

$HOME/.mh_profile	Defines the user's MH profile.
/usr/bin/mhpath	Contains the **mhpath** command.

Related Information

The **folder** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

migratelp Command

Purpose

Moves allocated logical partition from one physical partition to another physical partition on a different physical volume.

Syntax

migratelp *LVname*/*LPartnumber*[/*Copynumber*] *DestPV*[/*PPartNumber*]

Description

The **migratelp** moves the specified logical partition *LPartnumber* of the logical volume *LVname* to the *DestPV* physical volume. If the destination physical partition *PPartNumber* is specified it will be used, otherwise a destination partition is selected using the intra region policy of the logical volume. By default the first mirror copy of the logical partition in question is migrated. A value of 1, 2 or 3 can be specified for *Copynumber* to migrate a particular mirror copy.

> **Note:** You must consider the partition usage, reported by **lvmstat**, on the other active concurrent nodes in case of a concurrent volume group.

The **migratelp** command fails to migrate partitions of striped logical volumes.

Security

To use **migratelp**, you must have root user authority.

Examples

1. To move the first logical partitions of logical volume lv00 to hdisk1, type:
   ```
   migratelp lv00/1 hdisk1
   ```
2. To move second mirror copy of the third logical partitions of logical volume hd2 to hdisk5, type:
   ```
   migratelp hd2/3/2 hdisk5
   ```
3. To move third mirror copy of the 25th logical partitions of logical volume testlv to 100th partition of hdisk7, type:
   ```
   migratelp testlv/25/3 hdisk7/100
   ```

Files

/usr/sbin Directory where the **migratelp** resides.

Related Information

The **lslv** command, and **lvmstat** command.

The Logical Volume Storage Overview in the *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

migratepv Command

Purpose

Moves allocated physical partitions from one physical volume to one or more other physical volumes.

Syntax

migratepv [**-i**] [**-l** *LogicalVolume*] *SourcePhysicalVolume DestinationPhysicalVolume*...

Description

The **migratepv** command moves allocated physical partitions and the data they contain from the *SourcePhysicalVolume* to one or more other physical volumes. To limit the transfer to specific physical volumes, use the names of one or more physical volumes in the *DestinationPhysicalVolume* parameter; otherwise, all the physical volumes in the volume group are available for the transfer. All physical volumes must be within the same volume group. The specified source physical volume cannot be included in the list of *DestinationPhysicalVolume* parameters.

Notes:

1. To use this command, you must either have `root` user authority or be a member of the **system** group.
2. The **migratepv** command is not allowed on a snapshot volume group or a volume group that has a snapshot volume group.

The allocation of the new physical partitions follows the policies defined for the logical volumes that contain the physical partitions being moved.

The **migratepv** command (only when the source and target physical volumes are specified) fails when a boot logical volume is found on the source physical volume. When you migrate a physical volume, the boot logical volume must remain intact. Two contiguous physical partitions and the new boot image must be built on the new boot logical volume.

If you specify a logical volume that contains the boot image, the **migratepv -l** command attempts to find enough contiguous partitions on one of the target physical volumes. If the migration is successful, the **migratepv** command prints a message that recommends the user run the **bosboot** command to indicate a change in the boot device. The attempted migration fails if the **migratepv -l** command is unable to find enough contiguous space to satisfy the request.

Note: All Logical Volume Manager migrate functions work by creating a mirror of the logical volumes involved, then resynchronizing the logical volumes. The original logical volume is then removed. If the **migratepv** command is used to move a logical volume containing the primary dump device, the system will not have an accessible primary dump device during the execution of the command. Therefore, a dump taken during this execution may fail. To avoid this, reassign the primary dump device using the **sysdumpdev** command or ensure there is a secondary dump device defined before using **migratepv**.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit migratepv** fast path to run this command.

Note: For concurrent mode volume groups **migratepv** can only be used while active in enhanced concurrent mode or active in concurrent mode on SSA disks.

Flags

-i	Reads the *DestinationPhysicalVolume* parameter from standard input.
-l *LogicalVolume*	Moves only the physical partitions allocated to the specified logical volume and located on the specified source physical volume.

Examples

1. To move physical partitions from `hdisk1` to `hdisk6` and `hdisk7`, enter:

   ```
   migratepv hdisk1 hdisk6 hdisk7
   ```

   Physical partitions are moved from one physical volume to two others within the same volume group.
2. To move physical partitions in logical volume `lv02` from `hdisk1` to `hdisk6`, enter:

   ```
   migratepv  -l lv02 hdisk1 hdisk6
   ```

   Only those physical partitions contained in `lv02` are moved from one physical volume to another.

Files

/usr/sbin	Directory where the **migratepv** command resides.
/tmp	Directory where the temporary files are stored while the command is running.

Related Information

The **cplv** command, **lslv** command.

Migrating the Contents of a Physical Volume in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

System Dump Facility in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

---

mirrord Daemon

Purpose

Controls and monitors the mirror module for remote maintenance.

Syntax

mirrord

Description

The **mirrord** daemon controls and monitors the mirror module. The mirror daemon and mirror module work together to provide console mirroring, which is the two-way echoing of commands between the local operator's system console and a remote service expert's console. The local console, or BUMP console, is connected to line S1, and the remote console is connected to line S2 using a modem. The **mirrord daemon** is used to perform remote service support when the operating system is running.

> **Note:** To use remote service, the Software Error Logging and Dump Service Aids Package must be installed, remote service support must be valid, and the remote authorization flag must be set.

Normally, the **mirrord** daemon is started during the boot phase, just after the console configuration, but it can also be started from the command line. If the remote service support flag is not set (no remote service agreement) or if the remote authorization flag is not set, the daemon does nothing and exits.

If both flags are set, the daemon checks the key mode switch. If your machine has a key mode switch and it is in the Normal or Secure position, the daemon sleeps until the key is placed in the Service position. When the key is placed in the Service position, the daemon wakes up and checks the remote authorization flag and that the tty lines S1 and S2 (special files **/dev/ttyS1** and **/dev/ttyS2)** are managed by streams. If one of these checks fails, **mirrord** cannot perform console mirroring and returns a message explaining why not. If the checks pass, **mirrord** creates a lock file **/etc/locks/mirror**, kills processes belonging to line S2, pushes the mirror module, initializes line S2, and starts echoing in the mirror module. If your machine has the key mode switch, regardless of this switch, applications belonging to line S1 are never affected.

If your machine has a mode switch and it is already in the Service position and line S2 is connected when the daemon is started, the daemon simply pushes the mirror module (if necessary) and activates the echo mode. The daemon does not kill processes belonging to **/dev/ttyS2** in this case, since a remote service session may be underway. The modem used for the line S2 connection is configured according to the file **/usr/share/modems/mir_modem**, and its tty is configured according to the file **/usr/lib/mir_tty**. Normally, these files are installed by service personnel and do not need to be modified.

If the **mirrord** command is executed when the daemon is already installed, an error message is returned.

The **portmir** command (available in AIX 4.2.1 and later) can be used on most systems to mirror the console.

> **Note:** The **mirrord** command works only on multiprocessor systems with Micro Channel I/O. For IBM systems, this includes the IBM 7012 Model G Series, the IBM 7013 Model J Series, and the IBM 7015 Model R Series.

Signals

The daemon can be stopped using the **SIGTERM** or **SIGKILL** signals.

Examples

To start the daemon from the command line, simply enter:

```
mirrord
```

Files

/usr/lib/drivers/mirror	The mirror streams module.

/usr/lib/mir_tty	The tty configuration file for line S2.
/usr/share/modems/mir_modem	The modem configuration file for line S2.
/etc/locks/mirror	The **mirrord** lock file (exists when **mirrord** is active).
/dev/ttyS1 and **/dev/ttyS2**	The terminal special files controlled by **mirrord**.

Related Information

The **portmir** command.

---

mirrorvg Command

Purpose

Mirrors all the logical volumes that exist on a given volume group. This command only applies to AIX 4.2.1 or later.

Syntax

mirrorvg [**-S** | **-s**] [**-Q**] [**-c** *Copies*] [**-m**] *VolumeGroup* [*PhysicalVolume ...*]

Description

The **mirrorvg** command takes all the logical volumes on a given volume group and mirrors those logical volumes. This same functionality may also be accomplished manually if you execute the **mklvcopy** command for each individual logical volume in a volume group. As with **mklvcopy**, the target physical drives to be mirrored with data must already be members of the volume group. To add disks to a volume group, run the **extendvg** command.

By default, **mirrorvg** attempts to mirror the logical volumes onto any of the disks in a volume group. If you wish to control which drives are used for mirroring, you must include the list of disks in the input parameters, *PhysicalVolume.* Mirror strictness is enforced. Additionally, **mirrorvg** mirrors the logical volumes, using the default settings of the logical volume being mirrored. If you wish to violate mirror strictness or affect the policy by which the mirror is created, you must execute the mirroring of all logical volumes manually with the **mklvcopy** command.

When **mirrorvg** is executed, the default behavior of the command requires that the synchronization of the mirrors must complete before the command returns to the user. If you wish to avoid the delay, use the **-S** or **-s** option. Additionally, the default value of 2 copies is always used. To specify a value other than 2, use the **-c** option.

Notes:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. The **mirrorvg** command is not allowed on a snapshot volume group.

Attention: The **mirrorvg** command may take a significant amount of time before completing because of complex error checking, the amount of logical volumes to mirror in a volume group, and the time is takes to synchronize the new mirrored logical volumes.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit mirrorvg** fast path to run this command.

Flags

-c *Copies*
Specifies the minimum number of copies that each logical volume must have after the **mirrorvg** command has finished executing. It may be possible, through the independent use of **mklvcopy**, that some logical volumes may have more than the minimum number specified after the **mirrorvg** command has executed. Minimum value is 2 and 3 is the maximum value. A value of 1 is ignored.

-m *exact map*
Allows mirroring of logical volumes in the exact physical partition order that the original copy is ordered. This option requires you to specify a `PhysicalVolume(s)` where the exact map copy should be placed. If the space is insufficient for an exact mapping, then the command will fail. You should add new drives or pick a different set of drives that will satisfy an exact logical volume mapping of the entire volume group. The designated disks must be equal to or exceed the size of the drives which are to be exactly mirrored, regardless of if the entire disk is used. Also, if any logical volume to be mirrored is already mirrored, this command will fail.

-Q *Quorum Keep*
By default in **mirrorvg**, when a volume group's contents becomes mirrored, volume group quorum is disabled. If the user wishes to keep the volume group quorum requirement after mirroring is complete, this option should be used in the command. For later quorum changes, refer to the **chvg** command.

-S *Background Sync*
Returns the **mirrorvg** command immediately and starts a background **syncvg** of the volume group. With this option, it is not obvious when the mirrors have completely finished their synchronization. However, as portions of the mirrors become synchronized, they are immediately used by the operating system in mirror usage.

-s *Disable Sync*
Returns the **mirrorvg** command immediately without performing any type of mirror synchronization. If this option is used, the mirror may exist for a logical volume but is not used by the operating system until it has been synchronized with the **syncvg** command.

The following is a description of **rootvg**:

rootvg mirroring
When the **rootvg** mirroring has completed, you must perform three additional tasks: **bosboot, bootlist**, and **reboot**.

The **bosboot** command is required to customize the bootrec of the newly mirrored drive. The **bootlist** command needs to be performed to instruct the system which disk and order you prefer the mirrored boot process to start.

Finally, the default of this command is for Quorum to be turned off. For this to take effect on a **rootvg** volume group, the system must be rebooted.

non-rootvg mirroring
When this volume group has been mirrored, the default command causes Quorum to deactivated. The user must close all open logical volumes, execute **varyoffvg** and then **varyonvg** on the volume group for the system to understand that quorum is or is not needed for the volume group. If you do not **revaryon** the volume group, mirror will still work correctly. However, any quorum changes will not have taken effect.

rootvg and non-rootvg mirroring

The system dump devices, primary and secondary, should not be mirrored. In some systems, the paging device and the dump device are the same device. However, most users want the paging device mirrored. When **mirrorvg** detects that a dump device and the paging device are the same, the logical volume will be mirrored automatically.

If **mirrorvg** detects that the dump and paging device are different logical volumes, the paging device is automatically mirrored, but the dump logical volume is not. The dump device can be queried and modified with the **sysdumpdev** command.

Examples

1. To triply mirror a volume group, enter:

   ```
   mirrorvg -c 3 workvg
   ```

   The logical partitions in the logical volumes held on `workvg` now have three copies.
2. To get default mirroring of rootvg, enter:

   ```
   mirrorvg rootvg
   ```

   `rootvg` now has two copies.
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.
4. To sync the newly created mirrors in the background, enter:

   ```
   mirrorvg -S -c 3 workvg
   ```
5. To create an exact mapped volume group, enter:

   ```
   mirrorvg -m datavg hdisk2 hdisk3
   ```

Files

/usr/sbin Directory where the **mirrorvg** command resides.

Related Information

The **mklvcopy** command, **unmirrorvg** command, **syncvg** command, **extendvg** command, **reducevg** command, **sysdumpdev** command.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

mk_niscachemgr Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **nis_cachemgr** daemon and invokes the daemon by using the **startsrc** command.

Syntax

/usr/sbin/mk_niscachemgr [**-I**] | [**-B**] | [**-N**]

Description

The **mk_niscachemgr** command uncomments the entry in the **/etc/rc.nfs** file for the **nis_cachemgr** daemon. The **mk_niscachemgr** command starts the daemon by using the **startsrc** command.

> **Note:** The **mk_nisd**, **mk_cachemgr**, **mk_nispasswdd**, **rm_nisd**, **rm_cachemgr**, and **rm_nispasswdd** commands do two things:
>
> - Alter the entries of daemon startup calls in **/etc/rc.nfs**.
> - Alter the default behavior of the daemon **src** entities.
>
> For example, if the **rpc.nisd** daemon is supposed to start with the **-Y** flag, this will not be explicitly set in the **/etc/rc.nfs** entry for starting the **rpc.nisd** daemon. Instead, a **chssys** is executed to place the default options which are added (if any) to the daemons during startup. To verify that these options exist, use the **lssrc -S -s** *subsystem* command to show the default options.

Flags

-I Uncomments the entry in the **/etc/rc.nfs** file to start the **nis_cachemgr** daemon on the next system restart.

-B Uncomments the entry in the **/etc/rc.nfs** file to start the **nis_cachemgr** daemon and uses the **startsrc** command to start the **nis_cachemgr** daemon. This flag is the default.

-N Uses the **startsrc** command to start the **nis_cachemgr** daemon. This flag does not change the **/etc/rc.nfs** file.

Examples

To modify the **/etc/rc.nfs** file to invoke the **nis_cachemgr** daemon on the next system restart, enter:

```
mk_niscachemgr -I
```

Files

/etc/rc.nfs Contains the startup script for the NFS and NIS daemons.

Related Information

The **smit** command, **startsrc** command, and the **nis_cachemgr** daemon.

Network Information Services+ (NIS+) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How to Start and Stop the NIS+ Daemons in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide* and How to Export a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide.

---

mk_nisd Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **rpc.nisd** daemon and invokes the daemon by using the **startsrc** command.

Syntax

/usr/sbin/mk_nisd [**-l**] | [**-B**] | [**-N**] [**-s**] [**-y**] [**-b**]

Description

The **mk_nisd** command uncomments the entry in the **/etc/rc.nfs** file for the **rpc.nisd** daemon. The **mk_nisd** command starts the daemon by using the **startsrc** command.

> **Note:** The **mk_nisd**, **mk_cachemgr**, **mk_nispasswdd**, **rm_nisd**, **rm_cachemgr**, and **rm_nispasswdd** commands do two things:
>
> - Alter the entries of daemon startup calls in **/etc/rc.nfs**.
> - Alter the default behavior of the daemon **src** entities.
>
> For example, if the **rpc.nisd** daemon is supposed to start with the **-Y** flag, this will not be explicitly set in the **/etc/rc.nfs** entry for starting the **rpc.nisd** daemon. Instead, a **chssys** is executed to place the default options which are added (if any) to the daemons during startup. To verify that these options exist, use the **lssrc -S -s** *subsystem* command to show the default options.

Flags

-l Uncomments the entry in the **/etc/rc.nfs** file to start the **rpc.nisd** daemon on the next system-restart.

-B Uncomments the entry in the **/etc/rc.nfs** file to start the **rpc.nisd** daemon and uses the **startsrc** command to start the **rpc.nisd** daemon. This flag is the default.

-N Uses the **startsrc** command to start the **rpc.nisd** daemon. This flag does not change the **/etc/rc.nfs** file.

-s Starts the **rpc.nisd** with no DES authentication. If this flag is not used, the default **rpc.nisd** behavior is to always start with DES authentication. The **-s** option is used to make the **rpc.nisd** compatible with NIS(YP) clients.

-y Causes the **rpc.nisd** daemon to emulate a NIS(YP) service. This is not the default setting of **rpc.nisd** or **mk_nisd**.

-b Causes the **rpc.nisd** daemon to emulate the NIS(YP) DNS resolver service. This is not the default setting of **rpc.nisd** or **mk_nisd**.

> **Note:** The settings that result from using the **-a**, **-y**, and **-b** flags remain the default behavior of **rpc.nisd** after a system reboot if the **-l** or **-B** flags were used. The only way to restore settings is by executing **rm_nisd** and then executing **mk_nisd** once again.

Examples

1. To modify the **/etc/rc.nfs** file to invoke the **rpc.nisd** daemon on the next system-restart, enter:

   ```
   mk_nisd -I
   ```

2. To start the **rpc.nisd** daemon without DES authentication and to modify the **/etc/rc.nfs** file to invoke the **rpc.nisd** daemon without DES authentication upon reboot:

   ```
   mk_nisd -B -s
   ```

Files

/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Related Information

The **smit** command and the **startsrc** command.

The **rpc.nisd** daemon.

Network Information Services+ (NIS+) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How to Start and Stop the NIS+ Daemons in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide* and How to Export a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide.

---

mk_nispasswdd Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **rpc.nispasswdd** daemon and invokes the daemon by using the **startsrc** command.

Syntax

/usr/sbin/mk_nispasswdd [**-I**] | [**-B**] | [**-N**]

Description

The **mk_nispasswdd** command uncomments the entry in the **/etc/rc.nfs** file for the **rpc.nispasswdd** daemon. The **mk_nispasswdd** command starts the daemon using the **startsrc** command.

> **Note:** The **mk_nisd**, **mk_cachemgr**, **mk_nispasswdd**, **rm_nisd**, **rm_cachemgr**, and **rm_nispasswdd** commands do two things:
>
> - Alter the entries of daemon startup calls in **/etc/rc.nfs**.
> - Alter the default behavior of the daemon **src** entities.

For example, if the **rpc.nisd** daemon is supposed to start with the **-Y** flag, this will not be explicitly set in the **/etc/rc.nfs** entry for starting the **rpc.nisd** daemon. Instead, a **chssys** is executed to place the default options which are added (if any) to the daemons during startup. To verify that these options exist, use the **lssrc -S -s** *subsystem* command to show the default options.

Flags

-I Uncomments the entry in the **/etc/rc.nfs** file to start the **rpc.nispasswdd** daemon on the next system restart.

-B Uncomments the entry in the **/etc/rc.nfs** file to start the **rpc.nispasswdd** daemon and uses the **startsrc** command to start the **rpc.nispasswdd** daemon. The **-B** flag is the default.

-N Uses the **startsrc** command to start the **rpc.nispasswdd** daemon. The **-N** flag does not change the **/etc/rc.nfs** file.

Examples

1. To modify the **/etc/rc.nfs** file to invoke the **rpc.nispasswdd** daemon on the next system restart, enter:

```
mk_nispasswdd -I
```

Files

/etc/rc.nfs Contains the startup script for the NFS and NIS daemons.

Related Information

The **smit** command and **startsrc** command.

The **rpc.nispasswdd** daemon.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How to Start and Stop the NIS+ Daemons in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide* and How to Export a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

NIS+ Reference.

---

mkboot Command

Purpose

Creates the boot image, the boot record, and the service record. This command is *not* a user-level command and is NOT supported in AIX 4.2 or later.

Syntax

mkboot -d *Device* [**-b**] [**-D**] [**-c**] [**-h**] [**-i**] [**-l**] [**-l** *LVDev*] { **-k** *Kernel* | **-e** *Expander* } [**-L**] [**-s**] [**-r**] [**-p** *Offset*] [**-w**] **-f** *FileSystem*

Description

The **mkboot** command combines a kernel and file system into a boot image. The resulting image is written to standard out. It is copied to a boot device with the appropriate boot record information. The boot image can be made compressed or uncompressed and with or without a boot record at the beginning of the image. An image created for a tape is compressed with the boot record at the start of the image file. A disk boot image may be created without compression and has no boot record. The boot record is written to the first sector of the disk. The record contains information about the size and location of the image after it is written to the boot logical volume on that disk.

If the boot logical volume is mirrored, the **mkboot** command not only writes the boot image to each copy of the boot logical volume but also writes a boot record to each physical disk comprising the mirror. As long as the **mkboot** command is able to update at least one of the copies of a mirrored boot logical volume, no error is returned. To enable booting from each copy of a mirrored boot logical volume, each of the physical disks must be specified using the **bootlist** command. For more information regarding mirrored logical volumes, see Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

The **mkboot** command is usually called by the **bosboot** command. However, you can run the **mkboot** command a second time to put expand code at the beginning of a compressed boot image.

Flags

-b	Zeros out save-base fields. This flag is optional.
-d *Device*	Specifies the device required for the IPL record. This flag is required.
-c	Zeros out the boot record on the device. This flag is optional.
-D	Loads the low-level debugger at boot time.
-e *Expander*	Specifies kernel expansion code to create a compressed boot image file. Either the **-e** flag or the **-k** flag must be specified.
-f *FileSystem*	Specifies the boot file system. This flag is required.
-h	Prevents the **mkboot** command from updating the boot header. This flag is optional.
-i	Writes the normal portion of the boot record.
-l (uppercase i)	Invokes the low-level debugger at boot time.
-k *Kernel*	Specifies the kernel in the boot image. Either the **-k** flag or the **-e** flag must be specified.
-l (lowercase L) *LVDev*	Specifies the logical volume device that contains the loadable boot code.
-L	Enables lock instrumentation for MP systems. This flag has no effect on systems that are not using the MP kernel.
-p *Offset*	Specifies the address to use as `boot_pr_start` field in the boot record. This flag is used in creating the CD-ROM boot image. This flag is optional.
-r	Creates an image that is read-only storage (ROS) emulation code.
-s	Writes the service portion of the boot record.
-w	Outputs first two blocks of boot logical volume before the boot image. This flag is applicable to disk boot image only.

Security

Access Control: Only the root user can read and execute this command.

Examples

1. To create an uncompressed boot image, using the kernel **/usr/lib/boot/unix** and the **/tmp/bootfs** file system for the device **/dev/hdisk0**, enter

   ```
   mkboot -d /dev/hdisk0 -k /usr/lib/boot/unix -f /tmp/bootfs \
   -b -i -s > /tmp/boot.image
   ```

2. To clear the boot record but leave the PVID for disk **hdisk0**, enter:

   ```
   mkboot -d /dev/hdisk0 -c
   ```

3. Although the **mkboot** command combines a kernel and a random access memory (RAM) file system to create one boot image, you can run the **mkboot** command a second time to put expand code at the beginning of a compressed boot image. For example, enter:

   ```
   mkboot -b -d /dev/rmt0 -k unix -f ramfs | compress > /tmp/image
   mkboot -b -i -s -d /dev/rmt0 -k bootexpand -f /tmp/image \
   > bootfile
   ```

   for a bootable tape where:

`unix`	Specifies the kernel.
`ramfs`	Specifies the RAM disk file system.
`compress`	Specifies the compression or compact routine.
`bootexpand`	Specifies the expansion or kernel uncompact routine.

Files

/usr/include/sys/bootrecord.h	Specifies the structure of the boot record.

Related Information

The **bosboot** command, and **lockstat** command.

Understanding the Boot Process in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkC2admin Command

Purpose

Configure a system to operate in C2 Security Mode.

Syntax

mkC2admin { [**-m**] | [**-a** *address*] *hostname* }

Description

The **mkC2admin** command initializes the security directories for use in a C2 System configuration. The distributed database directories are created and symbolic links initialized. When a system is being configured as the Administrative Host (using the **-m** flag), an additional file system is created to hold the master copies of the administrative database files. Those files are stored in the directory **/etc/data.master** which has a logical volume name of **hd10sec**.

The administrative database files are divided into three categories. Those files that must be shared, those files that optionally may be shared, and those files that may not be shared. Optionally sharable files are described in the file **/etc/security/files.config**. That file consists of multiple lines of the format:

```
[y|n]|filename
```

and is editable by the administrator. To select an optionally sharable filename, the administrator sets the first field to the value **y**. To make an optionally sharable file be unshared, the field is set to the value **n**. All hosts in the C2 System must have an identical **/etc/security/files.config** file.

The system *hostname* must be defined in the **/etc/hosts** file at the time this command is run. If not, the IP address of the new C2 System Administrative Host may be provided with the **-a** option, and an entry will be added to **/etc/hosts**.

Flags

-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

hostname	Specifies the hostname.

Exit Status

0 The system has been properly configured to operate in the C2 mode.

1 The system was not installed with the C2 option.

2 The system could not be successfully configured to operate in C2 mode.

3 The system was previously configured to operate in C2 mode without having first been unconfigured.

Files

/usr/sbin/mkC2admin	Contains the mkC2admin command.

Related Information

The **chC2admin** command, **isC2host** command, **lsC2admin** command, **rmC2admin** command.

---

mkcatdefs Command

Purpose

Preprocesses a message source file.

Syntax

mkcatdefs *SymbolName SourceFile* ... [**-h**]

Description

The **mkcatdefs** command preprocesses a message source file for input to the **gencat** command.

The *SourceFile* message file contains symbolic identifiers. The **mkcatdefs** command produces the *SymbolName*_**msg.h** file, containing statements that equate symbolic identifiers with the set numbers and message ID numbers assigned by the **mkcatdefs** command.

The **mkcatdefs** command creates two outputs. The first is a header file called *SymbolName*_**msg.h**. You must include this *SymbolName*_**msg.h** file in your application program to associate the symbolic names to the set and message numbers assigned by the **mkcatdefs** command.

The **mkcatdefs** command sends message source data, with numbers instead of symbolic identifiers, to standard output. This output is suitable as input to the **gencat** command. You can use the **mkcatdefs** command output as input to the **gencat** command in the following ways:

- Use the **mkcatdefs** command with a **>** (redirection symbol) to write the new message source to a file. Use this file as input to the **gencat** command.
- Pipe the **mkcatdefs** command output file directly to the **gencat** command.
- Use the **runcat** command rather than the **mkcatdefs** command. The **runcat** command automatically sends the message source file through the **mkcatdefs** command and then pipes the file to the **gencat** command.

After running the **mkcatdefs** command, you can use symbolic names in an application to refer to messages.

Flags

-h Suppresses the generation of a *SymbolName*_**msg.h** file. This flag must be the last argument to the **mkcatdefs** command.

Examples

To process the `symb.msg` message source file and redirect the output to the `symb.src` file, enter:

```
mkcatdefs symb symb.msg > symb.src
```

The generated `symb_msg.h` file looks similar to the following:

```
#ifdef      _H_SYMB_MSG
#define     _H_SYMB_MSG
#include <limits.h>
#include <nl_types.h>
#define      MF_SYMB     "symb.cat"
/* The following was generated from symb.src. */
/* definitions for set MSFAC */
#define      SYM_FORM     1
#define      SYM_LEN      2
#define      MSG_H  6
#endif
```

The **mkcatdefs** command also creates the `symb.src` message catalog source file for the **gencat** command with numbers assigned to the symbolic identifiers:

```
$quote " Use double quotation marks to delimit message text
$delset 1
$set 1
1       "Symbolic identifiers can only contain alphanumeric \
characters or the _ (underscore character)\n"
2       "Symbolic identifiers cannot be more than 65 \
characters long\n"
5       "You can mix symbolic identifiers and numbers\n"
$quote
6       remember to include the "msg_h" file in your program
```

The assigned message numbers are noncontiguous because the source file contained a specific number. The **mkcatdefs** program always assigns the previous number plus 1 to a symbolic identifier.

Note: The **mkcatdefs** command inserts a **$delset** command before a **$set** command in the output message source file. This means you cannot add, delete, or replace single messages in an existing catalog when piping to the **gencat** command. You must enter all messages in the set.

Files

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

Related Information

The **dspcat** command, **dspmsg** command, **gencat** command, **runcat** command.

The **catclose** subroutine, **catgets** subroutine, **catopen** subroutine.

Using the Message Facility in *AIX 5L Version 5.2 National Language Support Guide and Reference*.

---

mkCCadmin Command

Purpose

Configure a system to operate in Common Criteria enabled Security Mode.

Syntax

mkCCadmin { [**-m**] | [**-a** *address*] *hostname* }

Description

The **mkCCadmin** command initializes the security directories for use in a Common Criteria enabled System configuration. The distributed database directories are created and symbolic links initialized. When a system is being configured as the Administrative Host (using the **-m** flag), an additional file system is created to hold the master copies of the administrative database files. Those files are stored in the directory **/etc/data.master** which has a logical volume name of **hd10sec**.

The administrative database files are divided into three categories. Those files that must be shared, those files that optionally may be shared, and those files that may not be shared. Optionally sharable files are described in the file **/etc/security/files.config**. That file consists of multiple lines of the format:

```
[y|n]|filename
```

and is editable by the administrator. To select an optionally sharable filename, the administrator sets the first field to the value **y**. To make an optionally sharable file be unshared, the field is set to the value **n**. All hosts in the Common Criteria enabled System must have an identical **/etc/security/files.config** file.

The system *hostname* must be defined in the **/etc/hosts** file at the time this command is run. If not, the IP address of the new Common Criteria enabled System Administrative Host may be provided with the **-a** option, and an entry will be added to **/etc/hosts**.

Flags

-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

hostname	Specifies the hostname.

Exit Status

0 The system has been properly configured to operate in the Common Criteria enabled mode.

1 The system was not installed with the Common Criteria enabled option.

2 The system could not be successfully configured to operate in Common Criteria enabled mode.

3 The system was previously configured to operate in Common Criteria enabled mode without having first been unconfigured.

Files

/usr/sbin/mkCCadmin	Contains the mkCCadmin command.

Related Information

The **chCCadmin** command, **isCChost** command, **lsCCadmin** command, **rmCCadmin** command.

---

mkcd Command

Purpose

Creates a multi-volume CD (or CDs) from a **mksysb** or **savevg** backup image.

Syntax

mkcd -r *directory* | **-d** *cd_device* | **-S** [**-m** *mksysb_image* | **-M** *mksysb_target* | **-s** *savevg_image* | **-v** *savevg_volume_group*] [**-C** *cd_fs_dir*] [**-I** *cd_image_dir*] [**-V** *cdfs_volume_group*] [**-B**] [**-p** *pkg_source_dir*] [**-R** | **-S**] [**-i** *image.data*] [**-u** *bosinst.data*] [**-e**] [**-P**] [**-I** *package_list*] [**-L**][**-b** *bundle_file*] [**-z** *custom_file*] [**-D**] [**-U**] [**-Y**]

Description

The **mckd** command creates a system backup image (**mksysb**) to CD-Recordable (CD-R) or DVD-Recordable (DVD-R, DVD-RAM) from the system **rootvg** or from a previously created **mksysb** image. It also creates a volume group backup image (**savevg**) to CD-R from a user-specified volume group or from a previously created **savevg** image.

For DVD media, system backups made with the **mkcd** command have a limitation in that they expect the media to be 4.7 GB or larger per side. The **mkcd** command will not process the next volume until it writes over 4 GB on the current volume, thus the use of smaller media would result in corruption when going beyond the media's capacity.

With the **mkcd** command, you can create bootable and non-bootable CDs in Rock Ridge (ISO9660) or UDF (Universal Disk Format) format.

See the **-L** flag for details about creating DVD-sized images. What applies to CDs also applies to DVDs, except where noted.

Note: The functionality required to create Rock Ridge format CD images and to write the CD image to the CD-R, DVD-R or DVD-RAM device is not part of the **mkcd** command. You must supply additional code to **mkcd** to do these tasks. The code will be called via shell scripts and then linked to **/usr/sbin/mkrr_fs** (for creating the Rock Ridge format image) and **/usr/sbin/burn_cd** (for writing to the CD-R device). Both links are called from the **mkcd** command.

Some sample shell scripts are included for different vendor-specific routines. You can find these scripts in **/usr/samples/oem_cdwriters**.

If you do not supply any file systems or directories as command parameters, **mkcd** creates the necessary file systems and removes them when the command finishes executing. File systems you supply are checked for adequate space and write access.

Note: If **mkcd** creates file systems in the backup volume group, they are excluded from the backup.

If you need to create multi-volume CDs because the volume group image does not fit on one CD, **mkcd** gives instructions for CD replacement and removal until all the volumes have been created.

Flags

-d *cd_device*
Indicates the CD-R, DVD-R or DVD-RAM device (`/dev/cd1`, for instance). This flag is required unless you use the **-S** flag.

-r *directory*
Indicates existing directory structure to burn onto a CD or DVD. This makes a CD image that is a copy of the given directory structure.

-m *mksysb_image*
Specifies a previously created **mksysb** image. If you do not give the **-m** flag **mkcd** calls **mksysb**. (See the **-M** flag for more information about where the **mksysb** image is placed.)

-s *savevg_image*
Indicates a previously created **savevg** image.See the Notes below.

-v *savevg_volume_group*
Denotes the volume group to be backed up using the **savevg** command. See the Notes below. (See the **-M** flag for more information about where the **savevg** image is placed.)

-C *cd_fs_dir*
Specifies the file system used to create the CD file system structure, which must have at least 645MB of available disk space (up to 4.38 GB for DVD sized images). The CD image will only consume as much room as necessary to contain all the data on the CD.

If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory exists, **mkcd** uses that directory. If you do not give the **-C** flag and the **/mkcd/cd_fs** directory does not exist, **mkcd** creates the file system **/mkcd/cd_fs** and removes it when the command finishes executing. The command creates the file system in the volume group indicated with the **-V** flag, or **rootvg** if that flag is not used. Each time you invoke the **mkcd** command, a unique subdirectory (using the process id) is created under the **/mkcd/cd_fs** directory, or in the directory specified with the **-C** flag.

Note: If performing DVD sized backups, the filesystems need to be *large file enabled*. This also requires setting the file **ulimit** size to *unlimited*.

-M *mksysb_target*
States the directory or file system where the **mksysb** or **savevg** image is stored if a previously created backup is not given with the **-m** or **-s** flags. If the **-M** flag is not used and a **mksysb** or **savevg image** is not provided, **mkcd** verifies that **/mkcd/***mksysb_image* exists. If the directory does not exist, then **mkcd** creates a separate file system, **/mkcd/***mksysb_image*, where the **mksysb** or **savevg** images are temporarily stored. The command creates the file system in the volume group given with the **-V** flag, or in **rootvg** if that flag is not used.

Note: If performing DVD sized backups, the filesystems need to be *large file enabled*. This also requires setting the file **ulimit** size to *unlimited*.

-I *cd_image_dir*

Specifies the directory or file system where the final CD images are stored before writing to the CD-R, DVD-R or DVD-RAM device. If this flag is not used, **mkcd** uses the **/mkcd/***cd_images* directory if it already exists. If not, the command creates the **/mkcd/***cd_images* file system in the volume group given with the **-V** flag, or in **rootvg** if that flag is not used.

If **mkcd** creates the file system, it is removed upon command completion, unless either the **-R** or **-S** flag is used. If the **-R** or **-S** flag is used, consideration must be made for adequate file system, directory, or disk space, especially when creating multi-volume CDs. The CD image will only consume as much room as necessary to contain all the data on the CD.Note: If performing DVD sized backups, the filesystems need to be *large file enabled*. This also requires setting the file **ulimit** size to *unlimited*.

-V *cdfs_volume_group*

Indicates the volume group used when creating the file systems needed for the **mkcd** command. If the **-V** flag is not given and a file system is needed but not there (because it was not supplied with other flags), then **rootvg** is the default volume group for creating the file systems. If **mkcd** creates the file systems in the backup volume group, those file systems are not included as part of the backup image. **mkcd**-created file systems are removed upon the command's completion.

-p *pkg_source_dir*

Names the directory or device that contains device and kernel package images. The device can only be a CD device (for example, `/dev/cd0`). If you use the same CD-R, DVD-R or DVD-RAM device that you gave with the **-d** flag, the product CD media must be inserted into the CD-R drive first. **mkcd** then prompts you to insert the writeable CD before the actual CD creation.

-B

Prevents **mkcd** from adding boot images (non-bootable CD) to the CD. Use this flag if creating a **mksysb** CD that you will not boot. Before installing the non-bootable **mksysb** CD you must boot a same level (V.R.M.) product CD. The **mkcd** command defaults to creating a bootable CD for the machine type of the source system. See the Notes below.

-R

Prevents **mkcd** from removing the final CD images. **mkcd** defaults by removing everything that it creates when it finishes executing. The **-R** flag allows multiple CD image sets to be stored, or for CD creation (burn) to occur on another system. If multiple volumes are needed, the final images are uniquely named using the process ID and volume suffixes.

-S

Stops **mkcd** before writing to the CD-R, DVD-R or DVD-RAM without removing the final CD images. The **-S** flag allows multiple CD sets to be created, or for CDs to be created on another system. The images remain in the directory marked by the **-I** flag, or in the **/mkcd/***cd_images* directory if the **-I** flag is not used. If multiple volumes are required, the final images are uniquely named using the process ID and volume suffixes.

-u *bosinst.data*

Specifies the user-supplied *bosinst.data* file. This data file takes precedence over the **bosinst.data** file in the **mksysb** image. If you do not give the **-u** flag, then **mkcd** restores **bosinst.data** from the given **mksysb** image, or generates a new **bosinst.data** file during the creation of **mksysb**.

-i *image.data*

Specifies the user-supplied *image.data* file. This data file takes precedence over the **image.data** file in the **mksysb** image. If you do not give the **-i** flag, then **mkcd** restores the **image.data** from the given **mksysb** image, or generates a new **image.data** file during the creation of **mksysb**.

> **Note:** The **-i** flag cannot be used to specify a user-supplied *vgname.data* file for use with a **savevg** image.

-e

Excludes the files and/or directories from the backup image listed in the **/etc/exclude.***volume_group* file. You cannot use this flag with the **-m** or **-s** flags.

-P Creates physical partition mapping during the **mksysb** or **savevg** creation. You cannot use this flag with the **-m** or **-s** flags.

-I *package_list* Specifies the file containing a list of additional packages you want copied to the **./usr/lpp/inst.images** directory of the CD file system. The images are copied from the location named with the **-p** flag. If you use the **-I** flag you must also use the **-p** flag.

-L Creates final CD images that are DVD sized (up to 4.38 GB).

-b *bundle_file* Gives the full pathname of the file containing a list of filesets to be installed after the **mksysb** is restored. This file is copied to **./usr/sys/inst.data/user_bundles/***bundle_file* in the CD file system and also copied to RAM in case the CD is unmounted. The file would be listed as BUNDLES=**/../usr/sys/inst.data/user_bundles/***bundle_file* in the **bosinst.data** file.

-z *custom_file* States the full pathname of the file to be copied to the root directory of the CD file system. This file could be a customization script specified in the **bosinst.data** file, such as `CUSTOMIZATION_FILE=filename`.

For example: If the file `my_script` is in **/tmp** on the machine where **mkcd** is running, then enter `-z/tmp/my_script` and specify `CUSTOMIZATION_FILE=my_script`. The code copies the script to the root directory of the RAM file system before it executes.

-D Turns on the debug output information feature. The default is no debug output.

-U Creates a **UDF** (Universal Disk Format) file system on DVD-RAM media. It does not require the amount of free space needed to create Rock Ridge format backups. It does not need the **/mkcd/cd_fs** and **/mkcd/cd_images** file systems. Therefore, the only temporary disk space it needs is to create the backup image that will be copied to the media. This means that the **-I** and **-C** flags do not apply to the **-U** flag. Because the backup is copied to the media, images cannot be created and burned later or on another system. So, the **-R** flag and **-S** flag do not apply when using the **-U** flag. You need to specify a device to write to with the **-d** flag. The **-U** flag does not use the **/usr/sbin/mkrr_fs** or **/usr/sbin/burn_cd** file systems.

-Y Accepts licenses.

Notes:

1. If you are creating a non-bootable CD (using the **-B** flag), you cannot use the **-p** or **-I** flags.
2. If you are creating a non-bootable CD with a **savevg** image (using the **-s** or **-v** flags), you cannot use the **-p**, **-I**, **-u**, **-i**, **-z**, or **-b** flags.

Examples

1. To generate a bootable system backup to the CD-R device named `/dev/cd1`, enter:
   ```
   mkcd -d /dev/cd1
   ```
2. To generate a system backup to the DVD-R or DVD-RAM device named `/dev/cd1`, enter:
   ```
   mkcd -d /dev/cd1 -L
   ```
3. To generate a non-bootable volume group backup of the volume group `myvg` to `/dev/cd1`, enter:
   ```
   mkcd -d /dev/cd1 -v myvg
   ```

   Note: All **savevg** backup images are non-bootable.
4. To generate a non-bootable system backup, but stop **mkcd** before the CD is created and save the final images to the `/mydata/my_cd` file system, and create the other **mkcd** filesystems in `myvg`, enter:
   ```
   mkcd -B -I /mydata/my_cd -V myvg -S
   ```
5. To create a CD or DVD that duplicates an existing directory structure

```
/mycd/a
/mycd/b/d
/mycd/c/f/g
```

use the following command:

```
mkcd -r /mycd -d /dev/cd1
```

After mounting with `mount -o ro /dev/cd1 /mnt`, **cd** to `/mnt`; a `find . -print` command displays:

```
./a
./b
./b/d
./c
./c/f
./c/f/g
```

Files

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

Related Information

The **mksysb** command and **savevg** command.

The **/image.data** file and the **/bosinst.data** file.

A procedure to verify the backup can be found in the article ″Verifying a System Backup″ in the *AIX 5L Version 5.2 Installation Guide and Reference*.

For more information about CD-R drives (including DVD-R and DVD-RAM drives) and CD-R creation software, refer to the following README file: `/usr/lpp/bos.sysmgt/README.oem_cdwriters`

---

mkcfsmnt Command

Purpose

Mounts a CacheFS directory.

Syntax

mkcfsmnt -d *PathName* **-t** { **nfs** | **cdrom**}] [**-h** *RemoteHost*] [**-p** { *RemoteDirectory* | *LocalDeviceName* }] [**-c** *CacheDirectory*] [**-o** *MountOptions*] [**-b** *BackingFileSystem*] [**-I** | **-B** | **-N**]

Description

The **mkcfsmnt** command constructs an entry that will be appended to the **/etc/filesystems** file, thus making a file system available for use as a cache file system. If the mount is to be permanent, this entry will remain. If the mount is temporary, the flags will be used directly for the **mount** command. CacheFS file systems are used to cache accesses to backing file systems. Backing file systems are generally NFS mounts.

Flags

-d *PathName*	Specifies the mount point for the cache directory.

-t	Selects file systems to be cached. **nfs** Specifies that the CacheFS file system is backed by an NFS mount. **cdrom** Specifies that the CacheFS file system is backed by a CDROM file system. (Currently not supported.)
-h *RemoteHost*	Specifies the NFS server that is exporting the directory.
-p *RemoteDirectory*	Specifies the directory that is mounted on the path name specified. This is commonly a remote file system that will be mounted via NFS or a local device name in the case of CDROM (Currently not supported.)
-c *CacheDirectory*	Specifies the location of the CacheFS file system. This must have been previously created by execution of the **cfsadmin** command.
-d *RemoteDirectory*	Specifies the directory that is mounted on the path name specified.
-o *MountOptions*	Specifies a comma-separated string of mount options that are dependent on the backing file system type. For instance, if it is NFS, the options would be those typically specified by the **-o** Options string to mount. See the **mount** command documentation for the acceptable values.
-b *BackingFileSystem*	Specifies a backing file system if it is already mounted. If this is not specified, then the command will do the mount itself on a temporary mount point. If this is not specified, then RemoteHost and RemoteDirectory must be specified.
-l	Causes an entry to be added to the **/etc/filesystems** file. The directory is not mounted.
-B	Adds an entry to the **/etc/filesystems** file and attempts to mount the file system. This flag is the default.
-N	Mounts the directory with the options specified, but does not modify the **/etc/filesystems** file.

Examples

To specify a CacheFS mount, type:

```
/usr/sbin/mkcfsmnt -t nfs -d /usr/share/man -p /usr/share/man -h host1 -c /cache/cache1 -o ro, intr -N
```

In this example, the **mkcfsmnt** command caches the remote directory **/usr/share/man** that resides on `host1` on the local **/usr/share/man** directory. The cache is kept in **/cache/cache1**, which was created with the **cfsadmin** command. CacheFS takes care of doing the NFS backing mount, because the **-b** flag has not been specified.

```
/usr/sbin/mkcfsmnt -t nfs -d /usr/share/man -p /usr/share/man -h host1 -c /cache/cache1 -b /backs/man -o ro, intr -N
```

In this example, the **mkcfsmnt** command caches the remote directory **/usr/share/man** residing on `host1` on the local **/usr/share/man** directory. The cache is kept in **/cache/cache1**, which was created with the **cfsadmin** command. The backing file system has already been mounted on **/backs/man**.

Files

/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

mkclass Command

Purpose

Create a Workload Management class.

Syntax

mkclass [**-a** *Attribute*=*Value ...*] [**-c** | **-m** | **-b** | **-C** | **-B** | **-P** | **-T** | **-L** | **-A** *KeyWord*=*Value*] [**-d** *Config_Dir*] [**-S** *SuperClass*] *Name*

Description

The **mkclass** command creates a superclass or a subclass identified by the *Name* parameter. The class must not already exist. The *Name* parameter can contain only upper and lower case letters, numbers and underscores. The name is in the format *supername* or *subname* (with the **-S** *supername* flag) or *supername.subname*. *Supername* and *subname* are each limited to 16 characters in length. The names **Default**, **System**, and **Shared** are reserved. They refer to predefined classes. Any *Attribute*=*Value* or *KeyWord*=**Value** argument initialise the specified attribute or resource limit. See **chclass** for more information. To set the process total limits (the limits that apply to each process of the class), use one or more of the options **-C** (totalCPU), **-B** (totalDiskIO) or **-A** (totalConnectTime), with the keyword value of hardmax. To set the class total limits (the limits that apply to the whole class), use one or more of the options **-P** (totalProcesses), **-T** (totalThreads) or **-L** (totalLogins), with the keyword value of hardmax. To reset any total limit, use "-" for *Value*. Process, class, or both total limits may be disabled when starting or updating the WLM (see **wlmcntrl** command).

Normally, **mkclass** adds the class and its attributes in the relevant WLM property files, and the modifications is applied to the in-core class definitions (active classes) only after an update of WLM using the **wlmcntrl** command.

If an empty string is passed as the configuration name (*Config_dir*) with the **-d** flag, the class is created only in the WLM in-core data structures, and no property file is updated, making the new class temporary (the change is lost if WLM is stopped and restarted or the system is rebooted).

Note: This command cannot apply to a set of time-based configurations (do not specify a set with the **-d** flag). If the current configuration is a set, the **-d** flag must be given to indicate which regular configuration the command should apply to.

Flags

-A hardmax=***Value*** Sets the maximum amount of time a login session in the class can stay active. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). As a user approaches this connection time limit, WLM will send a warning message to the session terminal. When the limit is reached, the user will be notified and the session leader will be sent the **SIGTERM** signal, and after a short grace period, the session will be terminated (**SIGKILL**).

-B hardmax=*Value* Sets the total amount of disk I/Os allowed for each process in the class. Value is specified as an integer, possibly appending the unit (KB for kilobytes, MB for megabytes, TB for terabytes, PB for petabytes, and EB for exabytes, default is kilobytes). After a process has used this amount of disk I/Os, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).

-C hardmax=*Value* Sets the total amount of CPU time allowed for each process in the class. Value is specified as an integer, possibly appending the unit (s for seconds, m for minutes, h for hours, d for days, and w for weeks, default is seconds). After a process has used this amount of time, the process will be sent the **SIGTERM** signal, and after a grace period, it will be killed (**SIGKILL**).

-d *Config_Dir* Use **/etc/wlm/***Config_Dir* as an alternate directory for the properties files. When this flag is not used, **mkclass** uses the configuration files in the directory pointed to by **/etc/wlm/current**. If an empty string is passed as the configuration name (`-d ""`) the new class is created only in the WLM in-core data structures and no configuration file is modified.

-L hardmax=*Value* Sets the total number of login sessions simultaneously available in the class. If a user tries to log onto the system and the login shell would end up in a class that has reached the total logins limit, the login operation will fail.

-P hardmax=*Value* Sets the maximum number of processes allowed in the class. If an operation would result in a new process entering the class when the class has this many processes in it, the operation will fail.

-S *SuperClass*
Specifies the name of the superclass when creating a subclass. There are two ways of creating the subclass **Sub** of superclass **Super**:

1. Specify the full name of the subclass as **Super.Sub** for *Name* and not use **-S**
2. Specify the **-S** flag to give the superclass name and use the short name for the subclass:

   ```
   mkclass options -S Super  Sub
   ```

-T hardmax=*Value*
Sets the maximum number of threads allowed in the class. If an operation would result in a new thread entering the class when the class has this many processes in it, the operation will fail. The total thread limit must be at least as large as the total process limit for a class. If a class has a total thread limit but no total process limit specified, the total process limit will be set to the total thread limit.

Security

Access control: Only the root user can create a superclass. Only root or authorized users whose user ID or group ID matches the user name or group name specified in the attributes **adminuser** and **admingroup** of a superclass can create a subclass of this superclass.

Files

classes	Contains the names and definitions of the classes.
limits	Contains the resource limits enforced each class.
shares	Contains the resource shares attributed to each class.

Related Information

The **wlmcntrl** command, **lsclass** command, **chclass** command, and **rmclass** command.

---

mkclient Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **ypbind** daemon and starts the **ypbind** daemon to configure a client.

Syntax

/usr/sbin/mkclient [**-I** | **-B** | **-N**] [**-S** ***server***]

Description

The **mkclient** command uncomments the entry to the **/etc/rc.nfs** file to start the **ypbind** daemon to configure a client. The **mkclient** command starts the **ypbind** daemon by using the appropriate System Resource Controller (SRC) command.

You can use the File Systems application in Web-based System Manager (wsm) to change file system characteristics.

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

Flags

-I Uncomments the entry for starting the **ypbind** daemon to the **/etc/rc.nfs** file. This entry causes the **ypbind** daemon to start during the next system restart.
-B Uncomments the entry to the **/etc/rc.nfs** file and starts the **ypbind** daemon. This flag is the default.
-N Causes the **startsrc** command to start the **ypbind** daemon. This flag does not affect the **/etc/rc.nfs** file.
-S Specifies which NIS *server* to use instead of broadcasting for one. This option must be used when no NIS server exists on the networks directly connected to the client machine.

Examples

To modify the **/etc/rc.nfs** file so that the **ypbind** daemon is started on the next system restart, enter:

```
mkclient -I
```

Files

/var/yp/domainname directory	Contains the NIS maps for the NIS domain.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Related Information

The **mkmaster** command, **rmyp** command, **smit** command, **startsrc** command.

The **ypbind** daemon, **yppasswdd** daemon, **ypserv** daemon, **ypupdated** daemon.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

System Resource Controller Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference.

---

mkcomg Command

Purpose

Creates a new communication group definition for a peer domain.

Syntax

mkcomg [**–s** *sensitivity*] [**–p** *period*] [**–t** *priority*] [**–x** {**br**}] [**–e** *NIM_path*] [**–m** *NIM_parameters*] [**–i n:***network_interface1*[**:***node1*] [**,***network_interface2*[**:***node2*]...] | **–S n:***"network_interface_selection_string"*] [**–h**] [**–TV**] *communication_group*

Description

The **mkcomg** command creates a new communication group definition for an online peer domain with the name specified by the *communication_group* parameter. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The communication group determines which devices are used for heartbeating in the peer domain. There can be more than one communication group in a peer domain.

The **mkcomg** command must be run on a node that is currently online in the peer domain where the communication group is to be defined. More than half of the nodes must be online to create a new communication group for the domain.

The **-e** and **-m** flags are used to set the network interface module (NIM) path and parameters. The NIM path is the path to the NIM that supports the adapter types used in the communication group. The NIM parameters are passed to NIM when it is started. If **-m** is not specified, the parameters predefined by topology services are used.

The communication group can be assigned to one or more interface resources. Use the **-i** flag to assign the communication group to a specific interface resource name. The interface resource can be limited to one on a particular node. An interface resource can also be specified using the **-S** flag and a selection string. This is used when specifying the interface resource name is not sufficient. The **-i** and **-S** flags cannot be used together. The **chcomg** command can also be used to assign a communication group to an interface resource.

Flags

-s *sensitivity*
: Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity value is an integer greater than or equal to 2. The default value is 4.

-p *period*
: Specifies the number of seconds between heartbeats. The period is an integer greater than or equal to 1. The default value is 1.

-t *priority*
: Specifies the priority. This value indicates the importance of this communication group with respect to others. It is used to order the heartbeat rings. The lower the number means the higher the priority. The highest priority is 1. The default value is 1 for IP networks and 255 for RS232 networks.

-x br
: Excludes controls for heartbeat mechanisms. This flag indicates that one or more controls for heartbeat mechanisms should not be used even if the underlying media support it. The following features can be excluded:

  b
  : Specifies that the broadcast feature should not be used even if the underlying media support it. If **-x b** is not specified, the broadcast feature will be used if the underlying media support it.

  r
  : Specifies that the source routing feature should not be used even if the underlying media support it. If **-x r** is not specified, the source routing feature will be used if the underlying media support it.

  To exclude more than one control, specify the feature characters consecutively: **-x br**.

-e *NIM_path*
: Specifies the network interface module (NIM) path name. This character string specifies the path name to the NIM that supports the adapter types in the communication group.

-m *NIM_parameters*
: Specifies the NIM start parameters. This character string is passed to the NIM when starting it.

-i n:*network_interface1***[:***node1***] [,***network_interface2***[:***node2***]...**
Assigns this communication group to the network interface resource defined by the network interface resource name and optionally the node name where it can be found.

If **-i** is specified, **-S** cannot be specified.

-S n:*"network_interface_selection_string"*
Assigns this communication group to the interface specified by the network interface selection string.

If **-S** is specified, **-i** cannot be specified.

-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

communication_group
Specifies the name of the new communication group that is to be created for the online peer domain. The name can contain any printable character.

Security

The user of the **mkcomg** command needs write permission for the **IBM.CommunicationGroup** resource class. Write permission for the **IBM.NetworkInterface** resource class is required to set the communication group for a network interface resource. By default, **root** on any node in the peer domain has read and write access to these resource classes 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 defined and online to the peer domain where the communication group is to be defined.

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

1. To define the communication group **ComGrp1** for the peer domain **ApplDomain** and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg ComGrp1
   ```

2. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, using a sensitivity of 1 and period of 3, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -s 1 -p 3 ComGrp1
   ```

3. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, not using broadcast, using a priority of 3, and **nodeA** is defined and online to **ApplDomain**, run this command on nodeA:

   ```
   mkcomg -x b -t 3 ComGrp1
   ```

4. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, not using broadcast, not using source routing, and **nodeA** is defined and online to **ApplDomain**, run the following command on **nodeA**:

   ```
   mkcomg -x br ComGrp1
   ```

5. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, using a NIM path of **/usr/sbin/rsct/bin/hats_nim**, NIM parameters **-l 5** to set the logging level, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -e /usr/sbin/rsct/bin/hats_nim -m "-l 5" ComGrp1
   ```

6. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, assign **ComGrp1** to the network interface resource named **eth0** on **nodeB**, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -i n:eth0:nodeB ComGrp1
   ```

7. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, assign **ComGrp1** to the network interface resource that uses the subnet 9.123.45.678, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -S n:"Subnet == 9.123.45.678" ComGrp1
   ```

Location

/usr/sbin/rsct/bin/mkcomg Contains the **mkcomg** command

Related Information

Commands: **lscomg**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmcomg**, **startrpdomain**

---

mkcondition Command

Purpose

Creates a new condition definition which can be monitored.

Syntax

mkcondition –r *resource_class* **–e** *event_expression* [**–E** *rearm_expression*] [**–d** *event_description*] [**–D** *rearm_description*] [**–m l** | **m** | **p**] [**–n** *node_name1*[**,***node_name2*...]] [**–s** *"selection_string"*] [**–p** *node_name*] [**–S c** |**w** |**i**] [**–h**] [**–TV**] *condition*

mkcondition –c *existing_condition*[**:***node_name*] [**–r** *resource_class*] [**–e** *event_expression*] [**–E** *rearm_expression*] [**–d** *event_description*] [**–D** *rearm_description*] [**–n** *node_name1*[**,***node_name2*...]] [**–p** *node_name*] [**–s***"selection_string"*] [**–m l** | **m** | **p**] [**–S c** | **w** | **i**] [**–h**] [**–TV**] *condition*

Description

The **mkcondition** command creates a new condition with the name specified by the *condition* parameter. The condition is used to monitor a resource for the occurrence of the condition (or event). One or more responses to an event can be defined by using the **mkresponse** command. The conditions can then be linked to the responses by using the **mkcondresp** command, or by using the **startcondresp** command to link the responses and start monitoring.

When in a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the condition definition. If the **-p** flag is not specified, the condition is defined on the local node. If the node where the condition will be defined is:

- in a cluster of nodes, the condition can monitor resources on more than one node. Use the **-n** flag to specify the nodes on which the condition will be monitored.
- the management server in a management domain, a management scope (**-m**) of local (**l**) or management domain (**m**) can be specified to indicate how the condition applies. The selection string will be evaluated using the entire management domain when management scope is set to the management domain and the node is the management server.
- a managed node in a management domain, only a management scope (**-m**) of local (**l**) can be used.
- in a peer domain, a management scope (**-m**) of peer domain (**p**) or local (**l**) can be used to indicate how the condition and the selection string apply.
- in both a management domain and a peer domain, a management scope (**-m**) of management domain (**m**), peer domain (**p**), or local (**l**) can be used to indicate how the condition and its selection string apply.

Flags

–c *existing_condition*[**:***node_name*]
: Copies an existing condition. The existing condition is defined on *node_name*. 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. If any other flags are specified, update the new condition as indicated by the flags. Links with responses are not copied.

–r *resource_class*
: Specifies the resource class to be monitored by this condition. The **lsrsrcdef** command can be used to list the resource class names.

–e *event_expression*
: Specifies the event expression, which determines when an event occurs. It includes a dynamic attribute of the *resource_class* with a mathematical comparison symbol (for example: >, <) and a constant. When this expression evaluates to TRUE, an event is generated.

–E *rearm_expression*
: Specifies the rearm expression. After the *rearm_expression* has evaluated to TRUE and an event is generated, the rearm expression determines when monitoring for the *event_expression* begins again. Typically, the rearm expression prevents multiple events from being generated for the same

event evaluation. The rearm expression includes a dynamic attribute of the *resource_class* with a mathematical comparison symbol (for example, >) and a constant.

–d *event_description*
Specifies user-supplied text that describes the event expression.

–D *rearm_description*
Specifies user-supplied text that describes the rearm expression.

–n *node_name1***[,***node_name2***...]**
Specifies the host name for a node (or a list of host names separated by commas for multiple nodes) where this condition will be monitored. You must specify the **-m** flag with a value of **m** or **p** if you want to use the **-n** flag. This way, you can monitor conditions on specific nodes instead of the entire domain.

–s *"selection_string"*
Specifies a selection string that is applied to all of the *resource_class* attributes to determine which resources should be monitored by the *event_expression*. The default is to monitor all resources within the *resource_class*. The resources used to evaluate the selection string is determined by the management scope (the **-m** flag). The selection string must be enclosed within double or single quotation marks. For information on how to specify selection strings, see the *RSCT Administration Guide*.

–S c | **w** | **i**
Specifies the severity of the event:

c Critical

w Warning

i Informational (for user reference). This is the default.

–m l | **m** | **p**
Specifies the management scope to which the condition applies. The management scope determines how the condition is registered and how the selection string is evaluated. The scope can be different from the current configuration, but monitoring cannot be started until an appropriate scope is selected. The valid values are:

l Specifies *local* scope. This is the default. The condition applies only to the local node (the node where the condition is defined; see the **-p** flag). Only the local node is used in evaluating the selection string.

m Specifies *management domain* scope. The condition applies to the management domain in which the node where the condition is defined belongs (see the **-p** flag). All nodes in the management domain are used in evaluating the selection string. The node where the condition is defined must be the management server in order to use management domain scope.

p Specifies *peer domain* scope. The condition applies to the peer domain in which the node where the condition is defined belongs (see the **-p** flag). All nodes in the peer domain are used in evaluating the selection string.

–p *node_name*
Specifies the name of the node where the condition is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

–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

condition The *condition* name is a character string that identifies the condition. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the **IBM.Condition** resource class to run **mkcondition**. 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.

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 define a condition with the name ″FileSystem space used″ to check for percentage of space used greater than 90% and to rearm when the percentage is back down below 85%, enter:

```
mkcondition -r IBM.FileSystem  \
-e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
"FileSystem space used"
```

2. To define a condition with the name ″tmp space used″ to check for percentage of space used greater than 90% for **/tmp** and to rearm when the percentage is back down below 85%, including comments, enter:

```
mkcondition -r IBM.FileSystem  \
-e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
-d "Generate event when tmp > 90% full"  \
-D "Restart monitoring tmp again after back down < 85% full"\
-s 'Name=="/tmp"'  "tmp space used"
```

3. To define a condition with the name ″Space used″ as a copy of ″FileSystem space used″, enter:

```
mkcondition -c "FileSystem space used"  "Space used"
```

4. To define a condition with the name ″var space used″ as a copy of ″tmp space used″, but change the selection to **/var**, enter:

```
mkcondition -c "tmp space used" -s 'Name=="/var"' \
"var space used"
```

These examples apply to management domains:

1. To define a condition with the name ″FileSystem space used″ to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n "*" -m d "FileSystem space used"
```

2. To define a condition with the name ″FileSystem space used″ to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes **nodeA** and **nodeB** in the domain, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m d \
"FileSystem space used"
```

3. To define a condition with the name ″nodeB FileSystem space used″ on **nodeB** to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the condition with local scope, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n "*" -m l -p nodeB \
"nodeB FileSystem space used"
```

4. To define a condition with the name ″local FileSystem space used″ to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node, run this command on a managed node:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l "local FileSystem space used"
```

These examples apply to peer domains:

1. To define a condition on **nodeA** with the name ″FileSystem space used″ to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m p -p nodeA "FileSystem space used"
```

2. To define a condition on **nodeC** with the name ″FileSystem space used″ to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes **nodeA** and **nodeB** in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m p -p nodeC \
"FileSystem space used"
```

3. To define a condition with the name ″local FileSystem space used″ on **nodeB** to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node only, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l "local FileSystem space used"
```

Location

/usr/sbin/rsct/bin/mkcondition
Contains the **mkcondition** command

Related Information

ERRM commands: **chcondition**, **lscondition**, **mkcondresp**, **mkresponse**, **rmcondition**, **startcondresp**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations and about how to use expressions and selection strings

---

mkcondresp Command

Purpose

Creates a link between a condition and one or more responses.

Syntax

mkcondresp [**–h**] [**–TV**] *condition*[**:***node_name*] *response1* [*response2*...]

Description

The **mkcondresp** command creates a link between a condition and one or more responses, but does not start monitoring. In a cluster environment, the condition and the response must be defined on the same node. You can start monitoring for this condition and its linked responses later by using the **startcondresp** command.

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.

Parameters

condition Specifies the name of the condition to be 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.

response1 **[***response2***...]**
Specifies one or more response names. All responses are linked to *condition*.

Security

The user needs write permission for the **IBM.Association** resource class to run **mkcondresp**. 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.

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 link the condition ″FileSystem space used″ to the response ″Broadcast event on-shift″, run this command:

   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To link the condition ″FileSystem space used″ to the responses ″Broadcast event on-shift″ and ″E-mail root any time″, run this command:

   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root any time"
   ```

These examples apply to management domains:

1. To link the condition ″FileSystem space used″ on the management server to the response ″Broadcast event on-shift″ (also on the management server), run this command on the management server:

   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To link the condition ″FileSystem space used″ on the managed node **nodeA** to the response ″Broadcastevent on-shift″, run this command on one of the nodes in the domain:

   ```
   mkcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To link the condition ″FileSystem space used″ on node **nodeA** to the response ″Broadcastevent on-shift″ (also on **nodeA**), run this command on one of the nodes in the domain:

   ```
   mkcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/mkcondresp
Contains the **mkcondresp** command

Related Information

ERRM commands: **lscondresp**, **mkcondition mkresponse**, **rmcondresp**, **startcondresp**, **stopcondresp**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations and about how to use expressions and selection strings

---

mkdev Command

Purpose

Adds a device to the system.

Syntax

mkdev { **-c** *Class* **-s** *Subclass* **-t** *Type* } [**-l** *Name*] [**-a** *Attribute*=*Value*] ... [**-d** | **-S** | **-R**] [**-f** *File*] [**-h**] [**-p** *ParentName*] [**-q**] [**-w** *ConnectionLocation*]

mkdev -l *Name* [**-h**] [**-q**] [**-S**]

Description

Attention: To protect the Configuration Database, the **mkdev** command is not interruptible. Stopping this command before execution is complete could result in a corrupted database.

The **mkdev** command either defines and makes available a device with the given device class (**-c** *Class* flag), type (**-t** *Type* flag), subclass (**-s** *Subclass* flag), connection location (**-w** *ConnectionLocation* flag), and the device logical name of the parent (**-p** *ParentName* flag), or makes available the previously defined

device specified by the given device logical name (**-l** *Name* flag). You can use any combination of the **-c**, **-s**, and **-t** flags needed to uniquely identify the predefined device.

If you specify the **-d** flag, the **mkdev** command only defines the device. If you specify the **-S** flag, the **mkdev** command brings the device to the Stopped state, if this state is supported, and does not make the device available. If you do not specify either the **-d** flag or the **-S** flag, the **mkdev** command makes the device available.

If you specify the **-R** flag, the **mkdev** command configures any parents of the specified device that are not already configured. Parents must be previously defined. The **-R** flag is not compatible with the **-d** and **-S** flags.

By using the **-l** flag with the **-c**, **-s**, and **-t** flags, you can specify the name you want the device to be known by. If you do not use the **-l** flag, a name will be automatically generated and assigned. Not all devices support user-supplied names.

Note: Queue device names must begin with an alphabetic character.

When using the **mkdev** command, you can supply the flags either on the command line or from the specified *File* parameter.

You can use the Devices application in Web-based System Manager (wsm) to change device characteristics.

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 mkdev** fast path to run this command.

Flags

-a *Attribute*=*Value*
: Specifies the device attribute value pairs to be used instead of the defaults. The *Attribute*=*Value* variable can be used to specify one attribute value pair or multiple attribute value pairs for one **-a** flag. If you use an **-a** flag with multiple attribute value pairs, the list of pairs must be enclosed in quotation marks with a blank space between the pairs. For example, entering `-a Attribute=Value` lists one attribute value pair per flag, while entering `-a 'Attribute1=Value1 Attribute2=Value2'` lists more than one attribute value pair. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.

-c *Class*
: Specifies the device class.

-d
: Defines the device in the Customized Devices object class. If you specify the **-d** flag, the **mkdev** command does not make the device available. This flag cannot be used with the **-S** flag.

-f *File*
: Reads the needed flags from the *File* parameter.

-h
: Displays the command usage message.

-l *Name*
: Specifies the already defined device, indicated by the *Name* variable, in the Customized Devices object class when not used with the **-c**, **-s**, and **-t** flags. The **-a**, **-p**, and **-w** flags cannot be used in this case. Queue device names must begin with an alphabetic character.

-p *ParentName*
: Specifies the device name, indicated by the *ParentName* variable, that you want assigned to the device when used with the **-c**, **-s**, and **-t** flags. Not all devices support this feature. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.

-q
: Suppresses the command output messages from standard output and standard error.

-R
: Configures any parents of the device that are not already configured. This flag is not compatible with the **-d** and **-S** flags.

-S	Prevents the device from being set to the Available state. This flag is only meaningful for those devices that support the Stopped state. This flag cannot be used with the **-d** flag.
-s *Subclass*	Specifies the subclass, indicated by the *Subclass* variable, of the device.
-t *Type*	Specifies the device type from the Predefined Devices object class.
-w *ConnectionLocation*	Specifies the connection location, indicated by the *ConnectionLocation* variable, on the parent. This flag cannot be used with the **-l** flag unless the **-c**, **-s**, and **-t** flags are also used.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Auditing Events:

Event	Information
DEV_Create	Method name, parameters
DEV_Configure	Errors
DEV_Start	Device name
DEV_Change	Parameters

Examples

1. To define (but not configure) a 4.0 GB 4mm Tape Drive tape drive connected to the SCSI adapter `scsi0` and using SCSI ID 5 and LUN of 0, type:

   ```
   mkdev -d -c tape -t4mm2gb -s scsi -p scsi0 -w 5,0
   ```

   The system displays a message similar to the following:

   ```
   rmt4 defined
   ```

2. To make the already defined tape device `rmt0` available to use, type:

   ```
   mkdev -l rmt0
   ```

   The system displays a message similar to the following:

   ```
   rmt0 available
   ```

3. To define and configure an RS-232 tty device connected to port 0 on the IBM 8-Port EIA-232/RS-422A (PCI) Adapter with the speed attribute set to 19200, and other attributes set from the `foo` file, type:

   ```
   mkdev -t tty -s rs232 -p sa3 -w 0 -a speed=19200 -f foo
   ```

   The system displays a message similar to the following:

   ```
   tty0 available
   ```

Files

/usr/sbin/mkdev	Contains the **mkdev** command.

Related Information

The **chdev** command, **lsattr** command, **lsconn** command, **lsdev** command, **lsparent** command, **rmdev** command.

Device Overview for System Management in the *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkdir Command

Purpose

Creates one or more new directories.

Syntax

mkdir [**-m** *Mode*] [**-p**] *Directory ...*

Description

The **mkdir** command creates one or more new directories specified by the *Directory* parameter. Each new directory contains the standard entries **.** (dot) and **..** (dot-dot). You can specify the permissions for the new directories with the **-m** *Mode* flag. You can use the **umask** subroutine to set the default mode for the **mkdir** command.

The owner-ID and group-ID of the new directories are set to the process's effective user-ID and group-ID, respectively. The setgid bit setting is inherited from the parent directory. To change the setgid bit, you can either specify the **-m** *Mode* flag or issue the **chmod** command after the creation of the directory.

Note: To make a new directory you must have write permission in the parent directory.

Flags

-m *Mode*

Sets the permission bits for the newly-created directories to the value specified by the *Mode* variable. The *Mode* variable takes the same values as the *Mode* parameter for the **chmod** command, either in symbolic or numeric form.

When you specify the **-m** flag using symbolic format, the op characters + (plus) and - (minus) are interpreted relative to the assumed permission setting a=rwx. The + adds permissions to the default mode, and the - deletes permissions from the default mode. Refer to the **chmod**command for a complete description of permission bits and formats.

-p

Creates missing intermediate path name directories. If the **-p** flag is not specified, the parent directory of each-newly created directory must already exist.

Intermediate directories are created through the automatic invocation of the following **mkdir** commands:

```
mkdir -p -m $(umask -S),u+wx $(dirname Directory) &&
mkdir [-m Mode] Directory
```

where the `[-m Mode]` represents any option supplied with your original invocation of the **mkdir** command.

The **mkdir** command ignores any *Directory* parameter that names an existing directory. No error is issued.

Exit Status

This command returns the following exit values:

0	All the specified directories were created successfully, or the **-p** option was specified and all the specified directories now exist.
>0	An error occurred.

Examples

1. To create a new directory called `Test` in the current working directory, enter:
   ```
   mkdir Test
   ```
   The `Test` directory is created with default permissions.
2. To create a new directory called `Test` with rwxr-xr-x permissions in the previously created `/home/demo/sub1` directory, enter:
   ```
   mkdir -m 755 /home/demo/sub1/Test
   ```
3. To create a new directory called `Test` with default permissions in the `/home/demo/sub2` directory, enter:
   ```
   mkdir -p /home/demo/sub2/Test
   ```
   The **-p** flag creates the `/home`, `/home/demo`, and `/home/demo/sub2` directories if they do not already exist.

Files

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

Related Information

The **chmod** command, **rm** command.

The **mkdir** subroutine, **umask** subroutine.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

File and Directory Access Modes in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* introduces file ownership and permissions to access files and directories.

Shells Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

mkdirhier Command

Purpose

Creates a hierarchy of directories or a single directory.

Syntax

mkdirhier *Directory* ...

Description

The **mkdirhier** command creates the specified directories. Unlike the **mkdir** command, if any of the parent directories of the specified directory do not exist, the **mkdirhier** command creates those directories as well as the specified directory.

Example

To create a directory named **foo2** or to create a hierarchy of directories named **foo, foo1**, and **foo2**, enter:

```
mkdirhier ~/foo/foo1/foo2
```

If **foo** and **foo1** already exist then the command creates **foo2**. However, if none of them exist then the command creates all three new directories.

Related Information

The **mkdir** command.

---

mkfifo Command

Purpose

Makes first-in-first-out (FIFO) special files.

Syntax

mkfifo [**-m** *Mode*] *File* ...

Description

The **mkfifo** command creates FIFO special files specified by the *File* parameter, in the order specified. If the **-m** *Mode* flag is not specified, the file mode of the FIFO file is the bitwise inclusive OR of the **S_IRUSR**, **S_IWUSR**, **S_IRGRP**, **S_IWGRP**, **S_IROTH**, and **S_IWOTH** permissions as modified by the file mode creation (see the **umask** command).

The **mkfifo** command functions similarly to the **mkfifo** subroutine.

Flags

-m *Mode* Sets the file permission bits of the newly created FIFO file to the specified mode values. The *Mode* variable is the same as the mode operand defined for the **chmod** command. The characters + (plus sign) and - (minus sign), if used, are interpreted relative to the initial value **a=rw** (that is, having permissions of **rw-rw-rw-**).

Exit Status

This command returns the following exit values:

0 All the specified FIFO special files were created successfully.
>0 An error occurred.

Examples

1. To create a FIFO special file with permissions `prw-r–r–`, enter:

   ```
   mkfifo  -m 644 /tmp/myfifo
   ```

   This command creates the `/tmp/myfifo` file with read/write permissions for the owner and read permission for the group and for others.
2. To create a FIFO special file using the - (minus sign) operand to set permissions of `prw-r—-`, enter:

   ```
   mkfifo  -m g-w,o-rw /tmp/fifo2
   ```

This command creates the /tmp/fifo2 file, removing write permission for the group and all permissions for others.

Note: If more than one file is created using the - (minus sign) operand, separate each mode specifier with a comma and no spaces.

Files

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

Related Information

The **chmod** command.

The **mkfifo** subroutine, **umask** subroutine.

---

mkfilt Command

Purpose

Activates or deactivates the filter rules.

Syntax

mkfilt -v 4 | **6** [**-d**] [**-u**] [**-z P** | **D**] [**-g** *start* | *stop*] [**-i**]

Description

Use the **mkfilt** command to activate or deactivate the filter rules. This command can also be used to control the filter logging function.

Flags

-v	IP version of the rules you want to activate. The value of **4** specifies IP version 4 and the value of **6** specifies IP version 6. The default (when this flag is not used) is to activate both IP version 4 and IP version 6. All the filter rules defined in the filter rule table for the IP version(s) will be activated or deactivated.
-d	Deactivates the active filter rules. This flag cannot be used with the **-u** flag.
-u	Activates the filter rules in the filter rule table. This flag cannot be used with the **-d** flag.
-z	Sets the action of the default filter rule to Permit (**P**) or Deny (**D**). The default filter rule is the last rule in the filter rule table that will apply to traffic that does not apply to any other filter rules in the table. Setting the action of this rule to Permit will allow all traffic that does not apply to any other filter rules. Setting this action to Deny will not allow traffic that does not apply to any other filter rules.
-g	This flag is used to either start (**start**) or stop (**stop**) the log functionality of the filter rule module.
-i	Initialization flag. This flag only applies when the **-u** flag is also used. If the **-i** flag is used, all the filter rules with an ″active″ status will be activated. If not used, all the filter rules in the filter rule table will be activated.

mkfont Command

Purpose

Adds a font path name to the Object Data Manager (ODM) that is loaded by the low function terminal (LFT) at boot time.

Syntax

mkfont [*FontPathName*]

Description

The **mkfont** command adds a fully qualified font file path name to the ODM. At boot time, the LFT loads the new font and any other fonts found in the ODM. The list of font information acquired by the LFT is passed to the default display device driver. The display driver selects from this list the font that best fits the display. If a default font was selected using the **chfont** command, the device driver uses that font.

Note: This command can be run only from an LFT.

You can use the Devices application in Web-based System Manager (wsm) to change device characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkfont** fast path to run this command.

The user must have root authority to execute this command.

Parameter

FontPathName The fully qualified pathname of a font file.

Examples

To add the font file **/usr/lpp/fonts/Rom10.snf**, enter the following command:

```
mkfont   /usr/lpp/fonts/Rom10.snf
```

Files

/bin/mkfont	Contains the **mkfont** command.
/usr/lpp/fonts	Contains the font directory.

Related Information

The **chfont** command, **lsfont** command.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

mkfontdir Command

Purpose

Creates a **fonts.dir** file from a directory of font files.

Syntax

mkfontdir [*DirectoryName* ...]

Description

The **mkfontdir** command creates a **fonts.dir** file from a directory of font files. For each directory argument, the **mkfontdir** command reads all of the bitmapped font files in the directory, searching for properties named FONT or the name of the file stripped of its suffix. These are used as font names, which are written to the **fonts.dir** file in the directory along with the name of the font file. The **fonts.dir** file is then used by the X server and the Font server to determine which fonts are available.

The kinds of font files read by the **mkfontdir** command depend upon the configuration parameters and typically include the following formats:

Portable Compile Format	(suffix **.pcf**)
Compressed PCF	(suffix **.pcf.Z**)
Server Natural Format	(suffix **.snf**)
Compressed SNF	(suffix **.snf.Z**)
Bitmap Distribution Format	(suffix **.bdf**)
Compressed BDF	(suffix **.bdf.Z**)

If a font exists in multiple formats, the most efficient format is used (PCF format before SNF then BDF formats).

Scalable fonts are not automatically recognized by **mkfontdir**. You can contruct a **fonts.scale** file (the format is identical to that in the **fonts.dir** file) containing entries for scalable fonts. Then, when you run **mkfontdir** on a directory, it copies entries from the **fonts.scale** file in that directory into the **fonts.dir** file it constructs in that directory.

You can create the **fonts.alias** file, which can be put in any directory of the font path, to map new names to existing fonts. This file should be edited by hand. The format is two columns separated by white space, with the first column containing aliases and the second column containing font-name patterns.

When a font alias is used by an X client, the X server searches for the name it references by looking through each font directory in turn. Therefore, the aliases and the font files do not need to be in the same directory.

To embed white space in aliases or font-name patterns, enclose them in double-quotation marks. To embed double-quotation marks, or any other characters, precede each character with a \ (backslash).

```
"magic-alias with spaces" "\"font\name\"with quotes"
regular-alias                   fixed
```

If the character string **FILE_NAMES_ALIASES** stands alone on a line, each file name in the directory when stripped of its suffix (such as **.pcf** or **.pcf.Z**) is used as an alias for that font.

The X server and the Font Server look for **fonts.dir** and **fonts.alias** files in each directory in the font path each time the font path is set.

Examples

To create a **fonts.dir** file from a directory of font files, enter:

```
mkfontdir DirectoryName
```

If no directory name is specified, the **mkfontdir** command reads the current directory.

Files

/usr/lib/X11/fonts	Is the directory containing font files, **fonts.dir** and **fonts.alias** files.

mkfs Command

Purpose

Makes a file system.

Syntax

mkfs [**-b** *Boot*] [**-l** *Label*] [**-i** *i-Nodes*] [**-o** *Options*] [**-p** *Prototype*] [**-s** *Size*] [**-v** *VolumeLabel*] [**-V** *VfsName*] *Device*

Description

The **mkfs** command makes a new file system on a specified device. The **mkfs** command initializes the volume label, file system label, and startup block.

The *Device* parameter specifies a block device name, raw device name, or file system name. If the parameter specifies a file system name, the **mkfs** command uses this name to obtain the following parameters from the applicable stanza in the **/etc/filesystems** file, unless these parameters are entered with the **mkfs** command:

dev	Device name
vol	Volume ID
size	File system size
boot	Program to be installed in the startup block
vfs	Definition of the virtual file system
options	File-system implementation-specific options of the form *Keyword*, *Keyword*=*Value*

Notes:

1. The file system is created with the setgid (set group ID) bit enabled. The setgid bit determines the default group permissions. All directories created under the new file system have the same default group permissions.
2. The **mkfs** command does not alter anything in a mounted file system, including the file system label. The file system label changes when you change the mount point, unless the file system is mounted.
3. For information about creating a filesystem on a striped logical volume, refer to “File Systems on Striped Logical Volumes” on page 524 in the **mklv** documentation.

Flags

-b *Boot*	Names the program to be installed in block 0 of the new file system.
-i *i-Nodes*	Specifies the initial number of i-nodes on the file system. This flag is ignored when creating a journaled file system.
-l *Label*	Specifies the file system label for the new file system.
-o *Options*	Specifies a comma-separated list of virtual file system implementation-specific options.

The following options are specific to the Journaled File System (JFS):

-o ag={ **8** | **16** | **32** | **64** }	Specifies the allocation group size in megabytes. An allocation group is a grouping of inodes and disk blocks similar to BSD cylinder groups. The default ag value is 8.
-o bf={ **true** | **false** }	Specifies a large file enabled file system. See ″Understanding Large File Enabled File Systems″ for more information. If you do not need a large file enabled file system, set this option to false; this is the default. Specifying **bf=true** requires a fragment size of 4096 and **compress=no**.
-o frag={ **512** | **1024** | **2048** | **4096** }	Specifies the JFS fragment size in bytes. A file system fragment is the smallest unit of disk storage that can be allocated to a file. The default fragment size is 4096 bytes.

-o compress={ **no** | **LZ** } Specifies data compression. If you do not want data to be compressed, set this option to no. Selecting compression requires a fragment size of 2048 or less.

-o nbpi={ **512** | **1024** | **2048** | **4096** |**8192** | **16384** | **32768** | **65536** | **131072** } Specifies the number of bytes per i-node (**nbpi**). The **nbpi** is the ratio of file system size in bytes to the total number of i-nodes. The default **nbpi** value is 4096 bytes. The values 32768, 65536, and 131072 only apply to AIX 4.2 or later.

Notes:

1. File systems created with an **ag** value greater than 8 is not recognized on an AIX 4.1 system.
2. The **ag**, **bf**, **compress**, **frag**, and **nbpi** attributes are set at file system creation and cannot be changed after the file system is successfully created. The **size** attribute defines the minimum file system size, and you cannot decrease it after the file system is created.
3. The root filesystem (**/**) cannot be compressed.
4. Some **nbpi** values and allocation group sizes are mutually exclusive. See ″Understanding JFS Size Limitations″ for information.

The following options are specific to the Enhanced Journaled File System (JFS2):

-o agblksize={ 512 | 1024 | 2048 | 4096 } Specifies the JFS2 block size in bytes. A file system block is the smallest unit of disk storage that can be allocated to a file. The default block size is 4096 bytes.

-o dense Specifies when files are created with holes JFS2 will allocate disk blocks for those holes and fill them with 0's.

-o name=*mountpoint* Specifies the mount point for the file system.

Specifies the log logical volume name. The specified logical volume is the logging device for the new JFS2.

-o log=INLINE Specifies to place the log in the logical volume with the JFS2 file system. The **INLINE** log will default to .4% of the logical volume size if logsize is not specified.

-o logsize=*Value* Specifies the size for an **INLINE** log in MBytes. Ignored if **INLINE** log not being used. Cannot be greater than 10% of the size of the filesystem.

Note: The **agblksize** and **dense** attributes are set at file system creation and cannot be changed after the file system is successfully created. The size attribute defines the minimum file system size, and you cannot decrease it after the file system is created.

-p *Prototype* Specifies the name of the prototype file. Options specified on the command line override attributes in the prototype file.

-s *Size* Specifies the size of the file system. Size can be specified in units of 512-byte blocks, Megabytes (suffix M should be used) or Gigabytes (suffix G should be used). See ″Understanding JFS Size Limitations″ for more information.

Notes:

1. The volume group in which the file system resides defines a maximum logical volume size and also limits the file system size.
2. The **-s** *Size* flag specifies the minimum file size and cannot be decreased after the file system has been successfully created.

-v *VolumeLabel* Specifies the volume label for the new file system.

-V *VfsName* Specifies the virtual file system (VFS) type. The VFS must have an entry in the **/etc/vfs** file.

Examples

1. To specify the volume and file system name for a new file system, type:

```
mkfs -lworks -vvol001 /dev/hd3
```

This command creates an empty file system on the /dev/hd3 device, giving it the volume serial number vol001 and file system name works. The new file system occupies the entire device. The file system has a default fragment size (4096 bytes) and a default nbpi ratio (4096).

2. To create a file system with nondefault attributes, type:

```
mkfs  -s 8192  -o nbpi=2048,frag=512 /dev/lv01
```

   This command creates an empty 4MB file system on the /dev/lv01 device with 512-byte fragments and 1 i-node for each 2048 bytes.

3. To create a large file enabled file system, type:

```
mkfs -V jfs -o nbpi=131072,bf=true,ag=64 /dev/lv01
```

   This creates a large file enabled JFS file system with an allocation group size of 64 megabytes and 1 inode for every 131072 bytes of disk. The size of the file system will be the size of the logical volume lv01.

4. To create a file system with nondefault attributes, type:

```
mkfs -s 4M -o nbpi=2048, frag=512 /dev/lv01
```

   This command creates an empty 4MB file system on the **/dev/lv01** device with 512-byte fragments and one i-node for each 2048 bytes.

Files

/etc/vfs	Contains descriptions of virtual file system types.
/etc/filesystems	Lists the known file systems and defines their characteristics.

Related Information

The **fsck** command, **mkproto** command, **proto** command.

The **ioctl** subroutine.

The **dir** file, **filesystems** file, **filsys.h** file.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Understanding JFS Size Limitations in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkgroup Command

Purpose

Creates a new group.

Syntax

mkgroup [**-R** *load_module*] [**-a**] [**-A**] [*Attribute*=*Value* ...] *Group*

Description

The **mkgroup** command creates a new group. The *Group* parameter must be a unique string of 8-byte or less and cannot be the **ALL** or **default** keywords. By default, the **mkgroup** command creates a standard group. To create an administrative group, specify the **-a** flag. You must be the root user or a user with GroupAdmin authorization to create an administrative group.

To create a group with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used to create the group. Load modules are defined in the **/usr/lib/security/methods.cfg** file.

You can use the Users application in Web-based System Manager (wsm) to change user characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkgroups** fast path to run this command.

Restrictions on Creating Group Names

To prevent login inconsistencies, you should avoid composing group names entirely of uppercase alphabetic characters. While the **mkgroup** command supports multi-byte group names, it is recommended that you restrict group names to characters with the POSIX portable filename character set.

To ensure that your user database remains uncorrupted, you must be careful when naming groups. Group names must not begin with a - (dash), + (plus sign), @ (at sign), or ~ (tilde). You cannot use the keywords **ALL** or **default** in a group name. Additionally, do not use any of the following characters within a group-name string:

:	Colon
″	Double quote
#	Pound sign
,	Comma
=	Equal sign
\	Back slash
/	Slash
?	Question mark
’	Single quote
`	Back quote

Finally, the *Name* parameter cannot contain any space, tab, or new-line characters.

Flags

-a	Creates an administrative group. Only the root user can use this flag.
-A	Sets the group administrator to the person who invoked the **mkgroup** command.
-R *load_module*	Specifies the loadable I&A module used to create the user.
Attribute=*Value*	Initializes a group with a specific attribute. See the **chgroup** command for more information about the group attributes.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. 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
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/mkuser.default**

Mode	File
x	/usr/lib/security/mkuser.sys

Auditing Events:

Event	Information
USER_Create	user

Limitations

Creating a group may not be supported by all loadable I&A modules. If the loadable I&A module does not support creating a group, an error is reported.

Examples

1. To create a new group account called `finance`, type:

   ```
   mkgroup finance
   ```

2. To create a new administrative group account called `payroll`, type:

   ```
   mkgroup -a payroll
   ```

   Only the root user can issue this command.

3. To create a new group account called `managers` and set yourself as the administrator, type:

   ```
   mkgroup -A managers
   ```

4. To create a new group account called `managers` and set the list of administrators to `steve` and `mike`, type:

   ```
   mkgroup adms=steve,mike managers
   ```

   The users `steve` and `mike` must already exist on the system.

5. To create a new group that is a LDAP I&A loadable module user, type:

   ```
   mkgroup -R LDAP monsters
   ```

Files

/usr/bin/mkgroup	Contains the **mkgroup** command.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/passwd	Contains basic user information.
/etc/security/passwd	Contains password information.

Related Information

The **chgroup** command, **chgrpmem** command, **chuser** command, **lsgroup** command, **lsuser** command, **mkuser** command, **passwd** command, **pwdadm** command, **rmgroup** command, **rmuser** command, **setgroups** command, **setsenv** command.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

mkhosts Command

Purpose

Generates the host table file.

Syntax

/usr/sbin/mkhosts [**-v**] *HostFile*

Description

The **mkhosts** command can be used to generate a hashed host database, using the filename specified by the *HostFile* parameter. It is not used if name resolution is performed by the **named** daemon. The host file is usually the **/etc/hosts** file, and in any case must be in the same format as the **/etc/hosts** file.

The **mkhosts** command generates database files named **hostfile.pag** and **hostfile.dir**. Updates to these files are built in a set of temporary files named **hostfile.new.pag** and **hostfile.new.dir**. The temporary files are copied into the database files only if the **hostfile.new.pag** and **hostfile.new.dir** files are built without errors.

The host file is used by one version of the **gethostbyaddr** and **gethostbyname** library routines for name resolution.

> **Note:** The version of the **gethostbyaddr** and **gethostbyname** library routines on this operating system do not support the **hostfile.pag** and **hostfile.dir** files.

After creating the host file, you can edit it to include the desired host entries.

Flags

-v Lists each host as it is added to the host file specified by the *HostFile* parameter.

Examples

Use the following command to generate the **/etc/hosts.pag** and **/etc/hosts.dir** files:

```
mkhosts /etc/hosts
```

This command creates two host files called **/etc/hosts.pag** and **/etc/hosts.dir**.

Files

hostfile.pag	One of two files containing the real database for name resolution.
hostfile.dir	One of two files containing the real database for name resolution.
hostfile.new.pag	One of two files containing the temporary database for name resolution.
hostfile.new.dir	One of two files containing the temporary database for name resolution.

Related Information

The **gettable** command, **htable** command.

The **named** daemon.

The **gethostbyname** subroutine, **gethostbyaddr** subroutine.

The **hosts** file format.

Naming in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

mkitab Command

Purpose

Makes records in the **/etc/inittab** file.

Syntax

mkitab [**-i** *Identifier*] { [*Identifier*] **:** [*RunLevel*] **:** [*Action*] **:** [*Command*] }

Description

The **mkitab** command adds a record to the **/etc/inittab** file. The *Identifier*:*RunLevel*:*Action*:*Command* parameter string specifies the new entry to the **/etc/inittab** file. You can insert a record after a specific record using the **-i** *Identifier* flag. The command finds the field specified by the *Identifier* parameter and inserts the new record after the one identified by the **-i** *Identifier* flag.

Parameters

The *Identifier*:*RunLevel*:*Action*:*Command* parameter string specifies the record in the **/etc/inittab** file, as follows:

Identifier A 14-character parameter that uniquely identifies an object. The *Identifier* must be unique. If the *Identifier* is not unique, the command is unsuccessful. The *Identifier* cannot be changed; if you try to change it, the command is unsuccessful.

RunLevel A 20-character parameter defining the run levels in which the *Identifier* can be processed. Each process started by the **init** command can be assigned one or more run levels in which it can be started.

Action A 20-character parameter that informs the **init** command how to process the *Command* parameter that you specify. The **init** command recognizes the following actions:

respawn
If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file.

wait When the **init** command enters the run level specified for this record, start the process and wait for it to stop. While the **init** command is in the same run level, all subsequent reads of the **/etc/inittab** file ignore this object.

once When the **init** command enters the run level specified for this record, start the process, do not wait for it to stop and when it does stop do not restart the process. If the system enters a new run level while the process is running, the process is not restarted.

boot Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Do not wait for the process to stop and when it does stop, do not restart the process. The run level for this process should be the default, or it must match the run level specified by the **init** command at startup time.

bootwait
Read this record only when the system boots and reads the **/etc/inittab** file. The **init** command starts the process. Wait for it to stop, and when it does stop, do not restart the process.

powerfail
Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal.

powerwait
Start the process identified in this record only when the **init** command receives a **SIGPWR** power fail signal, and wait until it stops before continuing to process the **/etc/inittab** file.

off If the process identified in this record is currently running, send the warning signal **SIGTERM** and wait 20 seconds before sending the **SIGKILL** kill signal. If the process is nonexistent, ignore this line.

hold When the process identified in this record is terminated, do not start a new one. The **hold** action can only be activated by the **phold** command.

ondemand
Functionally identical to **respawn**. If the process identified in this record does not exist, start the process. If the process currently exists, do nothing and continue scanning the **/etc/inittab** file. Specify this action to perform the **respawn** action when using **a**, **b**, or **c** run levels.

initdefault
A line with this action is processed only when the **init** command is originally invoked. The **init** command uses this line to determine which run level to originally enter. The command does this by taking the highest run level specified in the *RunLevel* parameter and using that as the command's initial state. If the *RunLevel* parameter is empty, its value is interpreted as 0123456789, and the **init** command enters a run level of **9**. If the **init** command does not find an **initdefault** line in the **inittab** file, it requests an initial run level from the operator at initial program load (IPL) time.

sysinit Start the process identified in this record before the **init** command tries to access the console. For example, you might use this to initialize devices.

Command A 1024-character field specifying the shell command.

Attention: To avoid possible corruption of system files, the **stdin**, **stdout**, and **stderr** files must be specified in the *Command* parameter with redirection, or they must be explicitly opened by the program being run by the command line.

Flags

-i *Identifier* Specifies which record in the **/etc/inittab** file the new record follows.

Examples

1. To add a new record to the **/etc/inittab** file, telling the **init** command to handle a login on tty2, type:
   ```
   mkitab "tty002:2:respawn:/usr/sbin/getty /dev/tty2"
   ```
2. To add a new record to the **/etc/inittab** file, telling the **init** command to execute the **/etc/rc.tcpip** file after the **/usr/sbin/srcmstr** file is started, type:
   ```
   mkitab -i srcmstr "rctcpip:2:wait:/etc/rc.tcpip > /dev/console"
   ```
3. To add a new record to the **/etc/inittab** file, telling the **init** command to execute the **/etc/rc** file and send its output to the boot log, type:
   ```
   mkitab ((rc:2:wait:/etc/rc 2>&1 | alog -tboot > /dev/console))
   ```

Files

/etc/inittab Contains the **mkitab** command.

Related Information

The **chitab** command, **lsitab** command, **rmitab** command, **init** command.

mkinstallp Command

Purpose

Creates software packages in **installp** format.

Syntax

mkinstallp [**-d** *BaseDirectory*] [**-T** *TemplateFile*]

Description

The **mkinstallp** command allows users to create their own software packages for AIX. Packages created with the **mkinstallp** command are in **installp** format and can be installed or removed with the **installp** command.

Files to be packaged by the **mkinstallp** command must be in a directory structure such that the location of the file relative to the root build directory is the same as the destination of the file after installation. For example, if **/usr/bin/***somecommand* is to be installed through a **mkinstallp** package, the *somecommand* parameter must be in the *buildroot***/usr/bin** directory when the **mkinstallp** command is run.

After the contents of a package are located in the correct directory structure, the **mkinstallp** command prompts for basic package data. This data includes the package name, requisites, descriptions of files to be packaged, and more. The **mkinstallp** command will then generate a template file based on responses given by the user. To prevent commnad-line prompting, template files can be created and edited directly by the user and passed to the **mkinstallp** command with the **-T** flag.

Flags

-d *BaseDirectory*	Specifies the root build directory containing the files to be packaged. If not specified, the current working directory is used.
-T *TemplateFile*	Specifies the full path name of the template file to be passed to the **mkinstallp** command. If not specified, the **mkinstallp** command prompts for package information and creates a new template file based on user responses.

Examples

This example demonstrates how to package the file **/usr/bin/foo** using the **/tmp/packages** directory as the root build directory.

First, create the directory structure by typing the following at the command line:

```
mkdir -p /tmp/packages/usr/bin
```

Then, type the following to create the file **/usr/bin/foo**:

```
touch /tmp/packages/usr/bin/foo
```

Then, type the following to create the package using the **mkinstallp** command:

```
mkinstallp -d /tmp/packages
```

For more examples, see the **/usr/lpp/bos/README.MKINSTALLP** file.

Files

/usr/sbin/mkinstallp	Contains the **mkinstallp** command.

Related Information

The **installp** command.

---

mkkeyserv Command

Purpose

Uncomments the entry in the **/etc/rc.nfs** file for the **keyserv** daemon and invokes the daemon by using the **startsrc** command.

Syntax

/usr/sbin/mkkeyserv [**-I** | **-B** | **-N**]

Description

The **mkkeyserv** command uncomments the entry in the **/etc/rc.nfs** file for the **keyserv** daemon. The **mkkeyserv** command starts the daemon by using the **startsrc** command.

You can use the File Systems application in Web-based System Manager (wsm) to change file system characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkkeyserv** fast path to run this command.

Flags

-I Uncomments the entry in the **/etc/rc.nfs** file to start the **keyserv** daemon on the next system restart.

-B Uncomments the entry in the **/etc/rc.nfs** file to start the **keyserv** daemon and uses the **startsrc** command to start the **keyserv** daemon. This flag is the default.

-N Uses the **startsrc** command to start the **keyserv** daemon. This flag does not change the **/etc/rc.nfs** file.

Examples

To modify the **/etc/rc.nfs** file to invoke the **keyserv** daemon on the next system restart, enter:

```
mkkeyserv -I
```

Files

/etc/rc.nfs Contains the startup script for the NFS and NIS daemons.

Related Information

The **smit** command, **startsrc** command.

The **keyserv** daemon.

System Management Interface Tool (SMIT) in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

How to Start and Stop the NIS Daemons in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide* and How to Export a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

---

mkkrb5clnt Command

Purpose

Configures a Kerberos client.

Syntax

mkkrb5clnt -h | [**-c** *KDC* **-r** *Realm* **-s** *Server* **-U** [**-a** *Admin*] **-d** *Domain* [**-A**] [**-i** *Database*] [**-K**] [**-T**]] [**-l** {*ldapserver* | *ldapserver:port*}]

Description

This command configures the Kerberos client. The first part of the command reads realm name, KDC, VDB path, and domain name from the input and generates a **krb5.conf** file.

/etc/krb5/krb5.conf:	Values for realm name, Kerberos admin server, and domain name are set as specified on the command line. Also updates the paths for **default_keytab_name**, **kdc**, and **kadmin** log files.

If DCE is not configured, this command creates a link to **/etc/krb5/krb5.conf** from **/etc/krb5.conf**.

The command also allows you to configure root as admin user, configure integrated Kerberos authentication, and configure Kerberos as default authentication scheme.

For integrated login, the **-i** flag requires the name of the database being used. For LDAP, use the load module name that specifies LDAP. For local files, use the keyword files.

Standard Output	Consists of information messages when the **-h** flag is used.
Standard Error	Consists of error messages when the command cannot complete successfully.

Flags

-a *Admin*	Specifies the principal name of the Kerberos server admin.
-A	Specifies root to be added as a Kerberos administrative user.
-c *KDC*	Specifies the KDC server.
-d *Domain*	Specifies the complete domain name for the Kerberos client.
-h	Specifies that the command is only to display the valid command syntax.
-i *Database*	Configures integrated Kerberos authentication.
-K	Specifies Kerberos to be configured as the default authentication scheme.

-l *ldapserver* / *ldapserver:port*	For servers, specifies the LDAP directory used to store the Network Authentication Service principal and policy information. For clients, specifies the LDAP directory server to use for Administration server and KDC discovery using LDAP. If the **-l** flag is used, then the KDC and server flags are optional. If the **-l** option is not used, the KDC and server flags must be specified. The port number can optionally be specified. For clients and servers, the port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections. **Note:** Only the client configuration is updated.
-r *Realm*	Specifies the full realm name for which the Kerberos client is to be configured.
-s *Server*	Specifies the fully qualified host name for Kerberos admin server.
-T	Specifies the flag to acquire server admin TGT based admin ticket.
-U	Undo the setup from the previous configuration command.

Exit Status

Failure of this command to execute successfully may result in incomplete client configuration.

0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

Only the root user is authorized to use this command.

Examples

1. To display the command syntax, type:

   ```
   mkkrb5clnt -h
   ```

2. To configure **testbox.austin.ibm.com** as a client to **sundial.austin.ibm.com** where KDC is also running on **sundial.austin.ibm.com**, type:

   ```
   mkkrb5clnt -c sundial.austin.ibm.com -r UD3A.AUSTIN.IBM.COM \
              -s sundial.austin.ibm.com -d austin.ibm.com
   ```

3. To configure **testbox.austin.ibm.com** as the client, make root as the server admin, configure integrated login, configure Kerberos as default authentication scheme, type:

   ```
   mkkrb5clnt -c sundial.austin.ibm.com -r UD3A.AUSTIN.IBM.COM \
        -s sundial.austin.ibm.com -d austin.ibm.com \
        -A -i files -K -T
   ```

Files

/usr/krb5/sbin	Contains the **mkkrb5clnt** command.

mkkrb5srv Command

Purpose

Configures a Kerberos server.

Syntax

mkkrb5srv -h | [**-r** *Realm* [**-s** *Server*] **-d** *Domain* **-a** *AdminName*] [**-l** *ldapserver* | *ldapserver:port*] [**-u** *ldap_DN*] [**-p** *ldap_DN_pw*] [**-f** {*keyring* | *keyring:entry_dn*}] [**-k** *keyring_pw*] [**-b** *bind_type*] [**-m** *masterkey_location*] [**-U**]

Description

The **mkkrb5srv** command configures the Kerberos server. This command creates the **kadm5.acl** file, the **kdc.conf** file, and the Kerberos database. It also adds the administrator to the database and updates the **/etc/inittab** file with Kerberos daemons. This command does the initial configuration once the variables are set. They can be modified by editing the following files:

/etc/krb5/krb5.conf:	Values for realm name, Kerberos admin server, and domain name are set as specified on the command line. Also updates the paths for **default_keytab_name**, **kdc**, and **kadmin** log files.
/var/krb5/krb5kdc/kdc.conf	This command sets the value for **kdc_ports**. Paths for database name, **admin_keytab**, **acl_file**, **dict_file**, **key_stash_file**. Values for **kadmin_port**, **max_life**, **max_renewable_life**, **master_key_type**, and **supported_enctypes**.
/var/krb5/krb5kdc/kadm5.acl	Sets up the acls for admin, root, and host principals.

If DCE is not configured, this command creates a link to **/etc/krb5/krb5.conf** from **/etc/krb5.conf**.

Standard Output	Consists of information messages when the **-h** flag is used.
Standard Error	Consists of error messages when the command cannot complete successfully.

Flags

-a *AdminName*	Specifies the Kerberos Principal name for the administrator.
-b *bind_type*	Specifies the LDAP bind type. Supported values are the following: • simple • cram-md5 • external These bind types can be specified in either upper case or lower case.
-d *Domain*	Specifies the domain name for the Kerberos realm.
-f {*keyring*	*keyring:entry_dn*}	Specifies the LDAP keyring database file name if you are using SSL communication.
-h	Specifies that the command is only to display the valid command syntax.
-k*keyring_pw*	Specifies the password for the LDAP keyring database file. If not specified, SSL uses the password that is encrypted in the appropriate password stash file.

-l *ldapserver* | *ldapserver:port*

For servers, specifies the LDAP directory used to store the Network Authentication Service principal and policy information.

For clients, specifies the LDAP directory server to use for Administration server and KDC discovery using LDAP. If the **-l** flag is used, then the KDC and server flags are optional. If the **-l** option is not used, the KDC and server flags must be specified. The port number can optionally be specified.

For clients and servers, the port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections.
Note: Only the client configuration is updated.

-m *masterkey_location*

Specifies the fully qualified file name for storing the master key in the local file system when using LDAP to store data.
Note: This flag is only for use with the LDAP directory.

-p *ldap_DN_pw*

Specifies the password for the entry being used for the *ldap_DN_pw*.

-r *Realm*

Specifies the realm for which the Kerberos server is to be configured.

-s *Server*

Specifies the fully qualified name of Kerberos Admin Server.

-u *ldap_DN*

Specifies the LDAP entry to be used as the *ldap_DN*.
Note: With external bind, the **-u** and **-p** flags are not required, and the values come form the certificate.

-U

Undo the setup from the previous configuration command.

Exit Status

Failure of this command to execute successfully results in incomplete server configuration.

0

Indicates the successful completion of the command.

1

Indicates that an error occurred.

Security

Only the root user is authorized to use this command.

Examples

1. To display the command syntax, type:
   ```
   mkkrb5srv -h
   ```
2. To configure **sundial** as a Kerberos server, type:
   ```
   mkkrb5srv -r UD3A.AUSTIN.IBM.COM -s sundial.austin.ibm.com -d austin.ibm.com
   ```

Files

/usr/sbin/mkkrb5srv

Contains the **mkkrb5srv** command.

mklost+found Command

Purpose
Creates a lost and found directory for the **fsck** command.

Syntax
mklost+found

Description
The **mklost+found** command creates a lost and found directory in the current directory. A number of empty files are created within the lost and found directory and then removed so that there are empty slots for the **fsck** command. The **fsck** command reconnects any orphaned files and directories by placing them in the lost and found directory with an assigned i-node number. The **mklost+found** command is not normally needed, since the **fsck** command automatically creates the lost and found directory when a new file system is created.

Examples
To make a lost+found directory for the **fsck** command, enter:
```
mklost+found
```

Files
/usr/sbin/mklost+found	Contains the **mklost+found** command.

Related Information
The **fsck** command, **mkfs** command.

The Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

The Files Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

mklv Command

Purpose
Creates a logical volume.

Syntax
mklv [**-a** *Position*] [**-b** *BadBlocks*] [**-c** *Copies*] [**-d** *Schedule*] [**-e** *Range*] [**-i**] [**-L** *Label*] [**-m** *MapFile*] [**-o** *Y / N*] [**-r** *Relocate*] [**-s** *Strict*] [**-t** *Type*] [**-u** *UpperBound*] [**-v** *Verify*] [**-w** *MirrorWriteConsistency*] [**-x** *Maximum*] [**-y** *NewLogicalVolume* | **-Y** *Prefix*] [**-S** *StripeSize*] [**-U** *Userid*] [**-G** *Groupid*] [**-P** *Modes*] *VolumeGroup Number* [*PhysicalVolume ...*]

Description
The **mklv** command creates a new logical volume within the *VolumeGroup*. For example, all file systems must be on separate logical volumes. The **mklv** command allocates the number of logical partitions to the new logical volume. If you specify one or more physical volumes with the *PhysicalVolume* parameter, only those physical volumes are available for allocating physical partitions; otherwise, all the physical volumes within the volume group are available.

The default settings provide the most commonly used characteristics, but use flags to tailor the logical volume to the requirements of your system. After a logical volume is created, its characteristics can be changed with the **chlv** command.

The default allocation policy is to use a minimum number of physical volumes per logical volume copy, to place the physical partitions belonging to a copy as contiguously as possible, and then to place the physical partitions in the desired region specified by the **-a** flag. Also, by default, each copy of a logical partition is placed on a separate physical volume.

The **-m** flag specifies exact physical partitions to be used when creating the logical volume.

If the *volume group* in which the logical volume is being created is in big vg format, **U**, **G**, and **P** flags can be used to set the ownership, group, and permissions respectively, of the special device files. Only root user will be able to set these values. If the *volume group* is exported, these values can be restored upon import if **R** flag is specified with **importvg** command.

You can specify logical volumes sizes in 512 Blocks/KB/MB/GB when using the **mklv** command. (See “Examples” on page 528.)

Physical partitions are numbered starting at the outermost edge with number one.

Notes:

1. Changes made to the logical volume are not reflected in the file systems. To change file system characteristics use the **chfs** command.
2. Each logical volume has a control block. This logical volume control block is the first few hundred bytes within the logical volume. Care has to be taken when reading and writing directly to the logical volume to allow for the control block. Logical volume data begins on the second 512-byte block.
3. To use this command, you must either have root user authority or be a member of the **system** group.
4. When creating a striped logical volume using the **-S** flag, you must specify two or more physical volumes or use the **-u** flag.
5. When creating a striped logical volume, the number of partitions must be an even multiple of the striping width.
6. To create a striped logical volume with more than one copy, all active nodes should be at least AIX 4.3.3 or later when the volume group is in the concurrent mode.
7. The **mklv** command is not allowed on a snapshot volume group.

You can use the Volumes application in Web-based System Manager to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit mklv** fast path to run this command.

File Systems on Striped Logical Volumes

If you want to create a filesystem on a striped logical volume, you should create the striped logical volume before you run the **crfs** command or **mkfs** command to create the filesystem. In order to maximize the use of disk space within the striping width, you should choose hard disks of the same size when creating the striped logical volume. The striping width is the number of hard disks that form the striped logical volume.

Flags

-a *Position*
Sets the intra-physical volume allocation policy (the position of the logical partitions on the physical volume). The *Position* variable can be one of the following:

- **m** Allocates logical partitions in the outer middle section of each physical volume. This is the default position.
- **c** Allocates logical partitions in the center section of each physical volume.
- **e** Allocates logical partitions in the outer edge section of each physical volume.
- **ie** Allocates logical partitions in the inner edge section of each physical volume.
- **im** Allocates logical partitions in the inner middle section of each physical volume.

-b *BadBlocks*
Sets the bad-block relocation policy. The *Relocation* variable can be one of the following:

- **y** Causes bad-block relocation to occur. This is the default.
- **n** Prevents bad-block relocation from occurring.

-c *Copies*
Sets the number of physical partitions allocated for each logical partition. The *Copies* variable can be set to a value from 1 to 3; the default is 1.

-d *Schedule*
Sets the scheduling policy when more than one logical partition is written. The *Schedule* variable can be one of the following:

- **p** Establishes a parallel scheduling policy. This is the default for scheduling policy.
- **ps** Parallel write with sequential read policy. All mirrors are written in parallel but always read from the first mirror if the first mirror is available.
- **pr** Parallel write round robbin read. This policy is similar to the parallel policy except an attempt is made to spread the reads to the logical volume more evenly across all mirrors.
- **s** Establishes a sequential scheduling policy.

-e *Range*
Sets the inter-physical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The *Range* value is limited by the *UpperBound* variable, (set with the **-u** flag) and can be one of the following:

- **x** Allocates across the maximum number of physical volumes.
- **m** Allocates logical partitions across the minimum number of physical volumes. This is the default range.

-G *Groupid*
Specifies group ID for the logical volume special file.

-i
Reads the *PhysicalVolume* parameter from standard input. Use the **-i** flag only when *PhysicalVolume* is entered through standard input.

-L
Sets the logical volume label. The default label is **None**. The maximum size of the label file is 127 characters.
Note: If the logical volume is going to be used as a journaled file system (JFS), then the JFS will use this field to store the mount point of the file system on that logical volume for future reference.

-m *MapFile*

Specifies the exact physical partitions to allocate. Partitions are used in the order given in the *MapFile* parameter. Used partitions in the *MapFile* parameter are not legal, because the new logical volume cannot occupy the same physical space as a previously allocated logical volume. All physical partitions belonging to a copy are allocated before allocating for the next copy of the logical volume. The *MapFile* parameter format is: `PVname:PPnum1[-PPnum2]`. In this example, `PVname` is a physical volume name (for example, `hdisk0`) as specified by the system. It is one record per physical partition or a range of consecutive physical partitions. `PPnum` is the physical partition number.

PVname Name of the physical volume as specified by the system.

PPnum Physical partition number.

-o*Y / N*

Turns on/off serialization of overlapping IOs. If serialization is turned on then overlapping IOs are not allowed on a block range and only a single IO in a block range is proccessed at any one time. Most applications like file systems and databases do serialization so serialization should be turned `off`. The default for new logical volumes is `off`.

-P *Modes*

Specifies permissions (file modes) for the logical volume special file.

-r *Relocate*

Sets the reorganization relocation flag. For striped logical volumes, the *Relocate* parameter must be set to **n** (the default for striped logical volumes). The *Relocate* parameter can be one of the following:

y Allows the logical volume to be relocated during reorganization. This is the default for relocation.

n Prevents the logical volume from being relocated during reorganization.

-s *Strict*

Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *Strict* parameter is represented by one of the following:

y Sets a strict allocation policy, so copies for a logical partition cannot share the same physical volume. This is the default for allocation policy.

n Does not set a strict allocation policy, so copies for a logical partition can share the same physical volume.

s Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror.

-S *StripeSize*

Specifies the number of bytes per striped. Must be a power of two, between 4K and 128K , for example 4K, 8K, 16K, 32K, 64K, or 128K.

Note: The **-d**, **-e**, and **-s** flags are not valid when creating a striped logical volume using the **-S** flag.

-t *Type*

Sets the logical volume type. The standard types are **jfs** (journaled file systems), **jfslog** (journaled file system logs), **jfs2** (enhanced journaled file system), **jfs2log** (enhanced journaled file system logs), and **paging** (paging spaces), but a user can define other logical volume types with this flag. You cannot create a striped logical volume of type **boot**. The default is **jfs**. If a log is manually created for a filesystem, the user must run the **logform** command to clean out the new **jfslog** before the log can be used. For example, to format the logical volume `logdev`, type:

```
logform /dev/logdev
```

where `/dev/logdev` is the absolute path to the logical volume.

-U *Userid*

Specifies user ID for logical volume special file.

-u *UpperBound*

Sets the maximum number of physical volumes for new allocation. The value of the *Upperbound* variable should be between one and the total number of physical volumes. When using striped logical volumes or super strictness the upper bound indicates the maximum number of physical volumes allowed for each mirror copy.

Note: When creating a super strict logical volume you must specify physical volumes or use the **-u** flag.

-v *Verify*

Sets the write-verify state for the logical volume. Causes (**y**) all writes to the logical volume to either be verified with a follow-up read, or prevents (**n**) the verification of all writes to the logical volume. The *Verify* parameter is represented by one of the following:

n Prevents the verification of all write operations to the logical volume. This is the default for the **-v** flag.

y Causes the verification of all write operations to the logical volume.

-w *MirrorWriteConsistency*

y or a Turns on *active* mirror write consistency that ensures data consistency among mirrored copies of a logical volume during typical I/O processing.

p Turns on *passive* mirror write consistency that ensures data consistency among mirrored copies during volume group synchronization after a system interruption.
Note: This functionality is only available on Big Volume Groups.

n No mirror write consistency. See the **-f** flag of the **syncvg** command.

-x *Maximum*

Sets the maximum number of logical partitions that can be allocated to the logical volume. The default value is 512. The number represented by the *Number* parameter must be equal to or less than the number represented by the *Maximum* variable. The maximum number of logical partitions per logical volume is 32,512.

-y *NewLogicalVolume*

Specifies the logical volume name to use instead of using a system-generated name. Logical volume names must be unique system wide name, and can range from 1 to 15 characters. If the *volume group* is varied on in concurrent mode, the new name should be unique across all the concurrent nodes the *volume group* is varied on. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices.

-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. The 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.

Examples

1. To make a logical volume in volume group `vg02` with one logical partition and a total of two copies of the data, type:

   ```
   mklv  -c 2 vg02 1
   ```

2. To make a logical volume in volume group `vg03` with nine logical partitions and a total of three copies spread across a maximum of two physical volumes, and whose allocation policy is not strict, type:

   ```
   mklv  -c 3   -u 2   -s n vg03 9
   ```

3. To make a logical volume in `vg04` with five logical partitions allocated across the center sections of the physical volumes when possible, with no bad-block relocation, and whose type is paging, type:

   ```
   mklv  -a c  -t paging  -b n vg04 5
   ```

4. To make a logical volume in `vg03` with 15 logical partitions chosen from physical volumes `hdisk5`, `hdisk6`, and `hdisk9`, type:

   ```
   mklv vg03 15 hdisk5 hdisk6 hdisk9
   ```

5. To make a striped logical volume in `vg05` with a stripe size of 64K across 3 physical volumes and 12 logical partitions, type:

   ```
   mklv -u 3 -S 64K vg05 12
   ```

6. To make a striped logical volume in `vg05` with a stripe size of 8K across hdisk1, hdisk2, and hdisk3 and 12 logical partitions, type:

   ```
   mklv -S 8K vg05 12 hdisk1 hdisk2 hdisk3
   ```

7. To request a logical volume with a minimum size of 10MB, type:

   ```
   mklv VGNAME 10M #
   ```

   The **mklv** command will determine the number of partitions needed to create a logical volume of at least that size.

   You can use uppercase and lowercase letters as follows:

   ```
   B/b       512 byte blocks
   K/k       KB
   M/m       MB
   G/g       GB
   ```

Files

/usr/sbin	Directory where the **mklv** command resides.
/tmp	Directory where the temporary files are stored while the command is running.
/dev	Directory where the character and block device entries for the logical volume are created.

Related Information

The **chfs** command, **chlv** command, **chpv** command, **extendlv** command, **mklvcopy** command, **rmlvcopy** command, **syncvg** command.

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mklvcopy Command

Purpose

Provides copies of data within the logical volume.

Syntax

mklvcopy [**-a** *Position*] [**-e** *Range*] [**-k**] [**-m** *MapFile*] [**-s** *Strict*] [**-u** *UpperBound*] *LogicalVolume Copies* [*PhysicalVolume*...]

Description

The **mklvcopy** command increases the number of copies in each logical partition in *LogicalVolume.* This is accomplished by increasing the total number of physical partitions for each logical partition to the number represented by *Copies*. The *LogicalVolume* parameter can be a logical volume name or logical volume ID. You can request that the physical partitions for the new copies be allocated on specific physical volumes (within the volume group) with the *PhysicalVolume* parameter; otherwise, all the physical volumes within the volume group are available for allocation.

The logical volume modified with this command uses the *Copies* parameter as its new **copy** characteristic. The data in the new copies are not synchronized until one of the following occurs: the **-k** option is used, the volume group is activated by the **varyonvg** command, or the volume group or logical volume is synchronized explicitly by the **syncvg** command. Individual logical partitions are always updated as they are written to.

The default allocation policy is to use minimum numbering of physical volumes per logical volume copy, to place the physical partitions belong to a copy as contiguously as possible, and then to place the physical partitions in the desired region specified by the **-a** flag. Also, by default, each copy of a logical partition is placed on a separate physical volume.

Notes:

1. To use this command, you must either have `root` user authority or be a member of the **system** group.
2. To create a copy of a striped logical volume, all active nodes using the volume group must be at least AIX 4.3.3 or later. Older versions will not be able to use the volume group after a mirror copy has been added to the striped logical volume.
3. The **mklvcopy** command is not allowed on a snapshot volume group.

You can use the Volumes application in Web-based System Manager (wsm) to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit mklvcopy** fast path to run this command.

Flags

Note: The **-e**, **-m**, **-s**, and **-u** flags are not valid with a striped logical volume.

-a *Position*
Sets the intra-physical volume allocation policy (the position of the logical partitions on the physical volume). The *Position* variable can be one of the following:

m Allocates logical partitions in the outer middle section of each physical volume. This is the default position.

c Allocates logical partitions in the center section of each physical volume.

e Allocates logical partitions in the outer edge section of each physical volume.

ie Allocated logical partitions in the inner edge section of each physical volume.

im Allocates logical partitions in the inner middle section of each physical volume.

-e *Range*
Sets the inter-physical volume allocation policy (the number of physical volumes to extend across, using the volumes that provide the best allocation). The *Range* value is limited by the *Upperbound* variable (set with the **-u** flag), and can be one of the following:

x Allocates across the maximum number of physical volumes.

m Allocates logical partitions across the minimum number of physical volumes. This is the default for the **-e** flag.

-k
Synchronizes data in the new partitions.

-m *MapFile*
Specifies the exact physical partitions to allocate. Partitions are used in the order given by the file designated by the *MapFile parameter.* Used partitions in the file are skipped. All physical partitions belonging to a copy are allocated before allocating for the next copy. The *MapFile* format is:

PVname:PPnum1[-PPnum2]
where *PVname* is a physical volume name (for example, `hdisk0`). It is one record per physical partition or a range of consecutive physical partitions.

PVname Name of the physical volume as specified by the system.

PPnum Physical partition number.

-s *Strict*
Determines the strict allocation policy. Copies of a logical partition can be allocated to share or not to share the same physical volume. The *Strict* variable is represented by one of the following:

y Sets a strict allocation policy, so copies for a logical partition cannot share the same physical volume. flag.

n Does not set a strict allocation policy, so copies for a logical partition can share the same physical volume.

s Sets a super strict allocation policy, so that the partitions allocated for one mirror cannot share a physical volume with the partitions from another mirror.

Note: When changing a nonsuper strict logical volume to a super strict logical volume, you must specify physical volumes or use the **-u** flag.

-u *UpperBound*
Sets the maximum number of physical volumes for new allocation. The value of the *Upperbound* variable should be between one and the total number of physical volumes. The default is the total total number of physical volumes in the volume group. When using striped logical volumes or super strictness the upper bound indicates the maximum number of physical volumes allowed for each mirror copy.

Example

To add physical partitions to the logical partitions in the logical volume `lv01`, so that a total of three copies exists for each logical partition, enter:

```
mklvcopy lv01 3
```

The logical partitions in the logical volume represented by directory `lv01` have three copies.

Files

/usr/sbin/mklvcopy	Contains the **mklvcopy** command.

Related Information

The **chlv** command, **lslv** command, **mklv** command, **syncvg** command, **varyonvg** command.

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkmaster Command

Purpose

Executes the **ypinit** command and starts the NIS daemons to configure a master server.

Syntax

/usr/sbin/mkmaster [**-s** *HostName* [**,***HostName* ...]] [**-O** | **-o**] [**-E** | **-e**] [**-P** | **-p**] [**-U** | **-u**] [**-C** | **-c**] [**-I** | **-B** | **-N**]

Description

The **mkmaster** command invokes the **ypinit** command to build the NIS maps for the current domain, if the domain name of the system is currently set. After the **ypinit** command completes successfully, the **mkmaster** command uncomments the entries in the **/etc/rc.nfs** file for the **ypserv** command, **yppasswdd** command, **ypupdated** command, and **ypbind** command.

You can use the File Systems application in Web-based System Manager (wsm) to change file system characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkmaster** fast path to run this command.

Flags

-s *HostName* [**,***HostName ...*]	Specifies the slave host names for this master server. These slave hosts must be configured after the master server has been configured. The **mkmaster** command automatically adds the current host to this list.
-O	Overwrites existing maps for this domain.
-o	Prevents the overwriting of existing maps for this domain. This flag is the default.
-E	Prevents further action if errors are encountered while building new maps. This is true for both the **ypinit** command and the **mkmaster** command. This flag is the default.
-e	Does not exit from the **ypinit** command and the **mkmaster** command if errors are encountered.
-P	Starts the **yppasswdd** daemon along with the **ypserv** daemon.
-p	Suppresses the start of the **yppasswdd** daemon. This flag is the default.

-U	Starts the **ypupdated** daemon along with the **ypserv** daemon.
-u	Suppresses the start of the **ypupdated** daemon. This flag is the default.
-C	Starts the **ypbind** daemon along with the **ypserv** daemon. This flag is the default.
-c	Suppresses the start of the **ypbind** daemon.
-I	Directs the **mkmaster** command to change the **/etc/rc.nfs** file to start the appropriate daemons on the next system restart. The execution of the **ypinit** command occurs when this command is invoked.
-B	Executes the **ypinit** command, uncomments the entries in the **/etc/rc.nfs** file, and starts the daemons. This flag is the system default.
-N	Executes the **ypinit** command and starts the appropriate daemons without changing the **/etc/rc.nfs** file.

Example

To execute the **ypinit** command, overwrite any existing maps for the current domain, and make `host1` and `host3` slave servers, enter:

```
mkmaster -s host1,host3 -O -p -u -B
```

This command will not start the **yppasswdd** daemon or the **ypupdated** daemon.

Files

/var/yp/*domainname* directory	Contains the NIS maps for the NIS domain.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Related Information

The **chmaster** command, **rmyp** command, **smit** command, **ypinit** command.

The **ypbind** daemon, **yppasswdd** daemon, **ypserv** daemon, **ypupdated** daemon.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

NIS Reference.

mknamsv Command

Purpose

Configures TCP/IP-based name service on a host for a client.

Syntax

mknamsv { **-a** *"Attribute=Value ..."* | **-A** *FileName* }

Description

The **mknamsv** high-level command configures a TCP/IP instance to use a name server. It calls the **namerslv** low-level command to configure the **resolv.conf** file appropriately.

You can use the Devices application in Web-based System Manager (wsm) to change device characteristics. You could also use the System Management Interface Tool (SMIT) **smit mknamerslv** fast path to run this command.

Flags

-A *FileName*
: Specifies the name of the file containing **named** daemon initialization information.

-a *"Attribute=Value..."*
: Specifies a list of attributes with corresponding values to be used for updating the **named** server initialization files in the database. Attributes available are:

  Domain
  : Domain name

  NameServer
  : Internet address of name server in dotted decimal format

Examples

1. To configure the name server initialization files, enter the command in the following format:

   ```
   mknamsv  -a"domain=austin.century.com nameserver=192.9.200.1"
   ```

   In this example the domain name and name server address are updated. The previous domain and name server are overwritten.
2. To configure name server initialization files according to information in another file, enter the command in the following format:

   ```
   mknamsv  -A namsv.file
   ```

   In this example, the file that contains the configuration information is `namsv.file`.

Files

/etc/resolv.conf Contains DOMAIN name server information for local resolver routines.

Related Information

The **namerslv** command.

Naming in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Configuring Name Servers in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The SMIT Interface for TCP/IP in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mknetid Command

Purpose

Generates data for the **netid.byname** map for use by the Network Inforamtion Services (NIS).

Syntax

To Generate raw **netid.byname** data:

/usr/sbin/makenetid [**-b**] [**-i** *NISInputFile*] [**-o** *NISOutputFile*] [**-d** *NISDomainName*] [**-m** *NISMasterName*] *InputFile OutputFile*

To Create an NIS Map:

/usr/sbin/mknetid [**-q**] [**-p** *PasswordFile*] [**-g** *GroupFile*] [**-h** *HostsFile*] [**-m** *NetidFile*]

Description

The **mknetid** command is used to produce the data for the **netid.byname** NIS map. It will parse the files specified on the command line and build the corresponding netid keys and values. Users will get the following entries:

```
unix.<uid>@<domainname> <uid>:<gid1>,<gid2>,...
```

Hosts will get the following entries:

```
unix.<hostname>@<domainname> 0:<hostname>
```

The domainname that is used is the same that is configured on the system at the time **mknetid** is run. The generated data is sent to **stdout**. Each line contains one entry, with the key and the data separated by a space.

Flags

-q	Quiet mode - do not report any warnings about the data.
-p *PasswordFile*	Specifies which passwd file to be used for reading the list of users.
-g *GroupFile*	Specifies which groups file to be used for reading the list of group memberships.
-h *HostsFile*	Specifies which hosts file to be used for reading the list of hostnames.
-m *NetidFile*	Specifies a file from which to read any additional netid entries to be included.

Files

/var/yp/Makefile	**mknetid** is most commonly used when rebuilding the NIS databases using **/var/yp/Makefile**.
/etc/passwd	
/etc/groups	
/etc/hosts	

Related Information

The **ypinit** command.

Network Information Services (NIS) Overview for System Management in AIX Version 4.3

Network Information Services (NIS and NIS+) Guide.

NIS Reference in AIX Version 4.3 Network Information Services (NIS and NIS+) Guide.

---

mknfs Command

Purpose

Configures the system to run NFS.

Syntax

/usr/sbin/mknfs [**-I** | **-N** | **-B**]

Description

The **mknfs** command configures the system to run the Network File System (NFS) daemons. The **mknfs** command adds an entry to the **inittab** file so that the **/etc/rc.nfs** file is executed on system restart.

Flags

-B Adds an entry to the **inittab** file to execute the **/etc/rc.nfs** file on system restart. The **mknfs** command also executes the **/etc/rc.nfs** file immediately to start the NFS daemons. This flag is the default.

-I Adds an entry to the **inittab** file to execute the **/etc/rc.nfs** file on system restart.

-N Starts the **/etc/rc.nfs** file to start the NFS daemons immediately. When started this way, the daemons run until the next system restart.

Files

inittab	Controls the initialization process of the system.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Related Information

The **chnfs** command, **mknfsexp** command, **mknfsmnt** command, **rmnfs** command.

NFS Installation and Configuration in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

List of NFS Commands.

mknfsexp Command

Purpose

Exports a directory to NFS clients.

Syntax

/usr/sbin/mknfsexp -d *Directory* [**-f** *Exports_File*] [**-t** [{ **rw** | **ro** | **rm -h** *HostName* [**,** *HostName ...*] }]] [**-a** *UID*] [**-r** *HostName* [**,** *HostName ...*]] [**-c** *HostName* [**,** *HostName ...*]] [**-s** | **-n**] [**-I** | **-B** | **-N**] [**-P** | **-p**]

Description

The **mknfsexp** command takes the flags and parameters specified and constructs a line that is syntactically correct for the **/etc/exports** file. If this command is invoked with the **-B** flag, an entry will be added to the **/etc/exports** file and the **exportfs** command is invoked to export the directory specified. Alternatively, the **-I** flag adds an entry to the **exports** file and does not export the directory, or the **-N** flag does not add an entry to the **exports** file but does export the directory.

Flags

-d *Directory*
Specifies the directory that is to be exported or changed.

-f *Exports_File*
Specifies the full path name of the exports file to use if other than the **/etc/exports** file.

-t *Type*
Specifies whether the directory is read-write, read-only, or read-mostly. The possible values for the *Type* variable are:

- **rw** Exports the read-write directory. This is the system default.
- **ro** Exports the read-only directory.
- **rm** Exports the read-mostly directory. If chosen, the **-h** flag must be used to specify the hosts that have read-write permission.

-h *HostName* [**,** *HostName*] ...
Specifies which hosts have read-write access to the directory. This option is valid only when the exported file is to be read-mostly.

-a *UID*
Uses the UID variable as the effective user ID only if a request comes from an unknown user. The default value of this option is -2.
Note: Root users (UID 0) are always considered unknown by the NFS server, unless they are included in the root option. Setting the value of UID to -1 disables anonymous access.

-r *HostName* [**,** *HostName*] ...
Gives root users on the specified hosts access to the directory. The default is for no hosts to be granted root access.

-c *HostName* [**,** *HostName*] ...
Gives mount access to each of the clients listed. A client can either be a host or a netgroup. The default is to allow all hosts access.

-s
Requires clients to use a more secure protocol when accessing the directory.

-n
Does not require the client to use the more secure protocol. This flag is the default.

-B
Adds an entry to the **/etc/exports** file and the **exportfs** command is executed to export the directory. This flag is the default.

-I	Adds an entry to the **/etc/exports** file so that the next time the **exportfs** command is run during system restart, the directory will be exported.
-N	Does not add an entry to the **/etc/exports** file but the **exportfs** command is run with the correct parameters so that the directory is exported.
-P	Specifies that the exported directory is to be a public directory. This flag only applies to AIX 4.2.1 or later.
-p	Specifies that the exported directory is not a public directory. This flag only applies to AIX 4.2.1 or later.

Examples

1. To export a directory with read-only permission, enter:

   ```
   mknfsexp -d /usr -t ro
   ```

   In this example, the `mknfsexp` command exports the `/usr` directory with read-only permission.
2. To export a directory with read-mostly permission and a secure protocol to specific hosts, enter:

   ```
   mknfsexp -d /home/guest -t rm -h bighost,littlehost -s
   ```

   In this example the `mknfsexp` command exports the `/home/guest` directory with read-mostly permission, using more secure protocol.
3. To export a directory with read-write permission to a specific netgroup and specific hosts, and to make the export effective on the next system restart, enter:

   ```
   mknfsexp -d /usr -t rw -c host1,host3,grp3 -I
   ```

   In the above example, the `mknfsexp` command exports the `/usr` directory and gives read and write permission to `host1`, `host2`, and `grp3`. The `-I` flag makes this change effective on the next system restart.
4. To export a directory with read-only permission to an exports file other than **/etc/exports**, enter:

   ```
   mknfsexp -d /usr -t ro -f /etc/exports.other
   ```

   In the above example, the `mknfsexp` command exports the `/usr` directory with read-only permission to the `/etc/exports.other` file.

Files

/etc/exports	Lists the directories that the server can export.

Related Information

The **chnfsexp** command, **exportfs** command, **rmnfsexp** command.

NFS Installation and Configuration, and Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

List of NFS Commands.

---

mknfsmnt Command

Purpose

Mounts a directory from an NFS server.

Syntax

/usr/sbin/mknfsmnt -f *PathName* **-d** *RemoteDirectory* **-h** *RemoteHost* [**-t** { **rw** | **ro** }] [**-m** *MountTypeName*] [**-w** { **fg** | **bg** }] [**-X** | **-x**] [**-S** | **-H**] [**-Y** | **-y**] [**-Z** | **-z**] [**-e** | **-E**] [**-a** | **-A**] [**-j** | [**-J**] [**-q** | [**-Q**] [**-g** | [**-G**] [**-s** | **-n**] [**-l** | **-B** | **-N**] [**-r** *TimesToRetry*] [**-R** *NumRetrans*] [**-b** *ReadBufferSize*] [**-c** *WriteBufferSize*] [**-o** *TimeOut*] [**-P** *PortNumber*] [**-u** *AcRegMin*] [**-U** *AcRegMax*] [**-v** *AcDirMin*] [**-V** *AcDirMax*] [**-T** *AcTimeOut*] [**-p** *NumBiods*] [**-K** { *any* | *2* | *3* }] [**-k** { *any* | *tcp* | *udp* }]

Description

The **mknfsmnt** command constructs an entry that will be appended to the **/etc/filesystems** file, thus making a file system available for mounting. If the mount is to be permanent, this entry will remain. If the mount is temporary, the flags will be used directly for the **mount** command. If the mount is soft, the system returns an error if the server does not respond. If the mount is hard, the client continues trying until the server responds. The hard mount is the default.

Flags

-A The **/etc/filesystems** entry for this file system will specify that it should be automatically mounted at system restart.

-a The **/etc/filesystems** entry for this file system will specify that it should not be automatically mounted at system restart. This is the default.

-B Adds an entry to the **/etc/filesystems** file and attempts to mount the file system. This flag is the default.

-b *ReadBufferSize* Indicates the size of the read buffer in bytes specified by the *ReadBufferSize* variable.

-c *WriteBufferSize* Indicates the size of the write buffer in bytes specified by the *WriteBufferSize* variable.

-d *RemoteDirectory* Specifies the directory that is mounted on the path name specified.

-E Allows keyboard interrupts on hard mounts.

-e Prevents keyboard interrupts on hard mounts. This is the default.

-f *PathName* Specifies the mount point for the remote directory.

-G Directs any file or directory created on the file system to inherit the group ID of the parent directory.

-g Does not direct new files or directories created on the file system to inherit the group ID of the parent directory. This is the default.

-H Creates a hard mount, which causes the client to continue retrying until the server responds. This is the default.

-h *RemoteHost* Specifies the NFS server that is exporting the directory.

-l Causes an entry to be added to the **/etc/filesystems** file. The directory is not mounted.

-J Indicates that **acl**s are used on this mount.

-j Indicates that **acl**s are not used on this mount. This is the default.

-K Specifies the NFS version used for this NFS mount. This flag only applies to AIX 4.2.1 or later. Options are:

any Uses the mount command to determine the correct match, first attempting the highest NFS version available.

2 Specifies NFS Version 2.

3 Specifies NFS Version 3.

-k Specifies the transport protocol used for the mount. This flag only applies to AIX 4.2.1 or later. Options are:

any Uses the mount command to select the protocol to use. TCP protocol is the preferred protocol.

tcp Specifies the TCP protocol.

udp Specifies the UDP protocol.

-m *MountTypeName* Specifies the type of file system to mount. File system types are specified in the **/etc/filesystems** file with the **type** variables. When the **mount -t** *MountTypeName* command is issued, all of the currently unmounted file systems with a type equal to the *MountTypeName* are mounted.

-N Mounts the directory with the options specified but does not modify the **/etc/filesystems** file.

-n Instructs the mount not to use a more secure protocol. This flag is the default.

-o *TimeOut* Indicates the length of the NFS timeout in tenths of a second as specified by the *TimeOut* variable.

-P *PortNumber* Indicates the Internet Protocol port number for the server.

-p *NumBiods* Specifies the number of **biod** daemons that are allowed to work on a particular file system. The **biod** daemons handle client requests and the default number of daemons is 6 (six).

-Q Requests that no posix pathconf information be exchanged and made available on an NFS Version 2 mount. Requires a mount Version 2 **rpc.mountd** at the NFS server.

-q Specifies that no posix pathconf information is exchanged if mounted as an NFS Version 2 mount. This is the default.

-r *TimesToRetry* Indicates the number of times to retry a mount. The default is 1000.

-R *NumRetrans* Specifies, for a soft mount, the number of times that a request is to be transmitted if it is not acknowledged by the server. If the request goes unacknowledged after *NumRetrans* transmissions, the client gives up on the request. If this flag is not specified, the default value of 3 is used.

-S Creates a soft mount, which means the system returns an error if the server does not respond.

-s Instructs the mount to use a more secure protocol.

-T *AcTimeOut* Sets minimum and maximum times allowed for regular files and directories to the number of seconds specified by the *Actimeo* variable. If this flag is specified, the other cached attribute times are overridden.

-t *Type* Specifies that the directory is either read-write or read-only.

rw Mounts the directory read-write. This type is the default for the system.

ro Mounts the directory read-only.

-U *AcRegMax* Holds cached attributes for no more than the number of seconds specified by the *AcRegMax* variable after file modification.

-u *AcRegMin* Holds cached attributes for at least the number of seconds specified by the *AcRegMin* variable after file modification.

-V *AcDirMax* Holds cached attributes for no more than the number of seconds specified by the *AcDirMax* variable after directory update.

-v *AcDirMin* Holds cached attributes for at least the number of seconds specified by the *AcDirMin* variable after directory update.

-w *Location* Indicates where the mount should be attempted. The *Location* variable can have one of the following values:

fg Attempts the mount in the foreground. This is the default value.

bg Attempts the mount in the background. If background is specified and the attempt to mount the directory fails, the mount will be retried in the background.

-x Specifies that the server does not support long device numbers.

Use this flag when mounting from an NFS server that does not correctly handle device numbers that are 32 bits long.

-X Specifies that the server does support long device numbers. This is the default.

-y Indicates that the execution of suid and sgid programs is not allowed in this file system.

-Y Indicates that the execution of suid and sgid programs are allowed in this file system. This is the default.

-z Indicates that device access through this mount is not allowed; that is, the device cannot be opened on this mount point.

-Z Indicates that device access through this mount is allowed. This is the default.

Example

To add the mount of a remote directory, enter:

```
mknfsmnt -f /usr/share/man -d /usr/share/man -h host1
```

In this example, the `mknfsmnt` command mounts the remote directory `/usr/share/man` on the `/usr/share/man` directory that resides on `host1`.

Files

/etc/filesystems Lists the remote file systems to be mounted during the system restart.

Related Information

The **chnfsmnt** command, **mount** command, **rmnfsmnt** command.

How to Mount a NFS File System Explicitly in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

List of NFS Commands.

---

mknod Command

Purpose

Creates a special file.

Syntax

Only executed by root or system group member

mknod *Name* { **b** | **c** } *Major Minor*

Creates FIFOs (named pipelines)

mknod *Name* { **p** }

Description

The **mknod** command makes a directory entry and corresponding i-node for a special file. The first parameter is the name of the *Name* entry device. Select a name that is descriptive of the device. The **mknod** command has two forms that have different flags.

The first form of the **mknod** command can only be executed by root or a member of the system group. In the first form, the **b** or **c** flag is used. The **b** flag indicates the special file is a block-oriented device (disk, diskette, or tape). The **c** flag indicates the special file is a character-oriented device (other devices).

The last two parameters of the first form are numbers specifying the *Major* device, which helps the operating system find the device driver code, and the *Minor* device, that is the unit drive or line number, which may be either decimal or octal. The major and minor numbers for a device are assigned by the

device's configure method and are kept in the CuDvDr class in ODM. It is important that major and minor numbers be defined in this object class to insure consistency of device definitions through the system.

In the second form of the **mknod** command, the **p** flag is used to create FIFOs (named pipelines).

Flags

b Indicates the special file is a block-oriented device (disk, diskette, or tape).
c Indicates the special file is a character-oriented device (other devices).
p Creates FIFOs (named pipelines).

Examples

To create the special file for a new diskette drive, enter:

```
mknod /dev/fd2  b 1 2
```

This creates the **/dev/fd2** special file that is a special block file with the major device number 1 and the minor device number 2.

Files

/usr/sbin/mknod Contains the **mknod** command.

Related Information

The **mknod** subroutine.

The List of Device Configuration Subroutines in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

---

mknotify Command

Purpose

Adds a notify method definition to the Notify object class.

Syntax

mknotify -n *NotifyName* **-m** *NotifyMethod*

Description

The **mknotify** command adds a notify method definition to the **Notify** object class. When a notify method is defined for both a subsystem name and a group name, the subsystem name takes precedence. For example, if the subsystem notify method is executed by the System Resources Controller (SRC), the group notify method is not performed.

The SRC places the name of the unsuccessful subsystem as the first argument to the method and the name of the unsuccessful subsystem group as the second argument.

Flags

-m *NotifyMethod* Specifies an absolute path to an executable program that starts when the subsystem stops abnormally.

-n *NotifyName* Specifies the subsystem or group name to which the notify method belongs. The *NotifyName* variable must exist as either a valid subsystem name or a valid group name in the **Subsystem** object class. The **mknotify** command is unsuccessful if the *NotifyName* variable already exists in the **Notify** object class.

Examples

1. To add a notify method for the `srctest` subsystem, enter:

   ```
   mknotify  -n srctest  -m /usr/lpp/srctest/failure
   ```

   This adds a subsystem notify method for the `srctest` subsystem, with a notify method designated in the `/usr/lpp/srctest/failure` file.
2. To add a notify method for the `tcpip` group, enter:

   ```
   mknotify  -n tcpip  -m /usr/lpp/tcpip/tcpfailure
   ```

   This adds a group notify method for the `tcpip` group, with a notify method designated in the `/usr/lpp/tcpip/tcpfailure` file.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC **Subsystem Configuration** object class.
/etc/objrepos/SRCnotify	Specifies the SRC **Notify Method** object class.

Related Information

The **rmnotify** command, **lssrc** command, **mkssys** command, **rmssys** command.

System Resource Controller Overview in the *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*

System Resource Controller (SRC) Overview for Programmers in the in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Defining Your Subsystem to the SRC in the in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

---

mkpasswd Command

Purpose

Organizes the basic user database for efficient searches.

Syntax

mkpasswd [**-v**] { **-f** | **-d** | **-c**} |*indexname*

Description

The **mkpasswd** generates indexes over certain security files. These indexes are used by the **getpwnam**, **getpwuid**, **getuserattr**, and **putuserattr** library subroutines.

This approach significantly enhances performance for large user base systems. The following indexes, defined in **/usr/include/usersec.h**, are created:

/etc/passwd.nm.idx:	Index over **/etc/passwd** file using username as key.
/etc/passwd.id.idx:	Index over **/etc/passwd** file using userid number as key.
/etc/security/passwd.idx:	Index over **/etc/security/passwd** file.
/etc/security/lastlog.idx:	Index over **/etc/security/lastlog** file.

Notes:

1. Modifying the security files over which indexes are built by an editor disables the use of indexing mechanism.
2. Indexed read of a data file is automatically done if a corresponding index exists over the file and is not older than it (except for lastlog index) .
3. In order for indexed mechanism to be used at login, the **mkpasswd** command must have generated indexes.
4. The indexing mechanism replaces the previous hashing mechanism which used dbm files.

Flags

-v	Reports progress if index built.
-f	Forces building of all indexes.
-d	Deletes all indexes.
-c	Checks all indexes and rebuilds the ones that look suspicious.
indexname	Forces building of a particular index.

Security

Access Control: Only the root user and members of the security group should have execute (x) access to this command. The command should be setuid to the root user so the command has access to the user database. Members of the security group should have access to all the files listed in the Files section. This command should have the **trusted computing base** attribute.

Files Accessed:

Mode	File
r	**/etc/passwd**
rw	**/etc/passwd.nm.idx***nnnn***.tmp** and **/etc/passwd.id.idx***nnnn***.tmp**
	where *nnnn* is the process id.
r	**/etc/security/passwd**
rw	**/etc/security/passwd.idx**
rw	**/etc/security/passwd.idx***nnnn***.tmp**
	where *nnnn* is the process id
r	**/etc/security/lastlog**
rw	**/etc/security/lastlog.idx**
rw	**/etc/security/lastlog.idx***nnnn***.tmp**
	where *nnnn* is the process id

Examples

1. To create and enable indexed read of security files, enter:
   ```
   mkpasswd -f
   ```
2. To create and enable indexed read of only the /etc/security/passwd file, enter:
   ```
   mkpasswd /etc/security/passwd.idx
   ```

3. To check and rebuild outdated or bad indexes, enter:

```
mkpasswd -c
```

Files

/usr/sbin/mkpasswd	Contains the mkpasswd command.
/etc/passwd	Contains basic user attributes.
/etc/security/passwd	Contains user password attributes
/etc/security/lastlog	Contains lastlog related attributes

Related Information

The **passwd** command, **pwdadm** command **mkuser** command **chuser** command **rmusers** command.

The **getpwnam** subroutine, **getpwuid** subroutine, **getuserattr** subroutine, **putuserattr** subroutine.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

---

mkpath Command

Purpose

Adds to the system another path to an MPIO capable device.

Syntax

mkpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-d**]

mkpath -h

Description

The **mkpath** command defines, and possibly configures, one or more paths to the target device (**-l** *Name*). The paths are identified by a combination of the **-l** *Name*, **-p** *Parent*, and **-w** *Connection* flags. Both the target device and parent must be previously defined in the system to define a path. They both must be ″AVAILABLE″ to configure a path.

If the **-d** flag is specified, the **mkpath** command only defines the new path definition to the system. If the **-d** flag is not specified, the **mkpath** command attempts to define the path, if it does not already exist, before it attempts to configure the path. Configuring a path requires the path to already be defined and both the device and the parent device to already be configured.

The **mkpath** command displays a status message upon completion. It is possible for some paths to configure and others to fail.

Note that any device that cannot be manually defined using the **mkdev** command will not be able to have paths manually defined to using the **mkpath** command. These limitations are due to the way that path information is stored for these devices. Fiber channel devices fall into this category.

The **mkpath** command provides status messages about the results of operation. Messages in one of the following formats will be generated:

path [available | defined]
This message is displayed when **mkpath** is run on a single path. If the path is successfully

configured the message ″path available″ is displayed. If the path is not successfully configured and there is no explicit error code returned by the method, the message ″path defined″ is displayed.

paths available
This message is displayed if multiple paths were identified and all paths were successfully configured.

some paths available
This message is displayed if multiple paths were identified, but only some of them were successfully configured.

no paths processed
This message is generated if no paths were found matching the selection criteria.

Flags

-d	Defines a new path to a device by adding a path definition to the system. The new path will not automatically be configured when the **-d** flag is specified. Note that only one path may be defined at a time.
-h	Displays the command usage message.
-l *Name*	Specifies the logical device name of the target device to which the path(s) are being added. The path(s) to be added are qualified by the **-p** and **-w** flags.
-p *Parent*	Indicates the logical device name of the parent device associated with the path(s) to be added. This flag is required if the **-d** flag is specified.
-w *Connection*	Indicates the connection information associated with the path to be added. This flag is required if the **-d** flag is specified.

Security

Privilege Control: Only the **root** user and members of the **system** group have execute access to this command.

Auditing Events:

Event	Information
DEV_Change	mkpath,Define,<define method arguments>
DEV_Change	mkpath,Configure,<configure method arguments>

Examples

1. To define and configure an already defined path between scsi0 and the hdisk1 device at SCSI ID 5 and LUN 0 (i.e., connection 5,0), enter:

   ```
   mkpath -l hdisk1 -p scsi0 -w 5,0
   ```

   The system displays a message similar to the following:

   ```
   path available
   ```

2. To configure an already defined path from '**fscsi0**' to fiber channel disk '**hdisk1**', the command would be:

   ```
   mkpath -l hdisk1 -p fscsi0
   ```

   The message would look similar to:

```
path available
```

3. To only add to the Customized Paths object class a path definition between **scsi0** and the **hdisk1** disk device at SCSI ID 5 and LUN 0, enter:

   ```
   mkpath -d -l hdisk1 -p scsi0 -w 5,0
   ```

   The system displays a message similar to the following:

   ```
   path defined
   ```

Files

/usr/sbin/mkpath	Contains the **mkpath** command.

Related Information

The **chpath** command, **lspath** command, **rmpath** command.

---

mkproto Command

Purpose

Constructs a prototype file system.

Syntax

mkproto *Special Prototype*

Description

The **mkproto** command is used to construct a prototype for a new file system. It exists solely for Berkeley Software Distribution (BSD) compatibility.

The *Special* parameter can be a block device name, raw device name, or file system name. The *Prototype* parameter is the name of the prototype file that specifies the structure and contents of the file system to be created. The **mkproto** command calls the **mkfs** command with the *Prototype* and *Special* parameters.

Prototype Files

The **mkproto** and **mkfs** commands require an extended prototype file to create a Journaled File System (JFS). A *prototype file* is a formatted listing of the contents and structure of a file system. A prototype file describes the file system by a series of tokens separated by spaces and new lines. The main body of a prototype file defines the objects of the file system.

A JFS prototype file consists of the main body, which can be created by the **proto** command, preceded by five special tokens. These five tokens are defined as follows:

1st token	Name of a file to be copied onto block 0 as the bootstrap program or the special token `<noboot>`.
2nd token	Size of the file system. For a JFS, the size is expressed in units of 512-byte blocks. If the 2nd token is 0, the **mkfs** command creates the file system to fill the entire logical volume.
3rd token	Number of i-nodes on the file system. This token is not used by a JFS but must be provided to preserve the position.
4th token	Size of the file system fragment in bytes. If the 4th token is 0 (zero), the **mkfs** command uses the default fragment size. For JFS, the token must be either 0 (default value used), 512, 1024, 2048, or 4096. The default fragment size is 4096 for a JFS. An invalid fragment size causes the **mkfs** command to fail.
5th token	Number of bytes per i-node (nbpi). If this token is 0, the **mkfs** command uses the default nbpi. For a JFS, this token must be either 0 (default value used), 512, 1024, 2048, 4096, 8192, or 16384. The default number of bytes per i-node is 4096 for a JFS. An invalid nbpi causes the **mkfs** command to fail.

The remaining tokens define the contents and structure of the file system. These tokens are grouped into sets, with each set defining one object of the file system. The syntax of each set is as follows:

{ [*Name*] { - | **d** | **b** | **c** | **l** | **L** | **p** } { **-** | **u** } { **-** | **g** } { **-** | **t** } *Mode Owner Group* { *Major Minor* | *SourceFile* | *DirectoryListing* } } | { **$** }

where:

Name
Specifies the name of the object as it is to appear in the new file system. The *Name* token is required for every object except for the root directory definition.

{ **-** | **d** | **b** | **c** | **l** | **L** | **p** } { **-** | **u** } { **-** | **g** } { **-** | **t** }
Represents a string of 4 positional characters, where:

{ - | d | b | c | l | L | p }
Defines the object type. Valid types are:

- **-** Regular file
- **d** Directory
- **b** Block special file
- **c** Character special file
- **l** Symbolic link
- **L** Hard link
- **p** Named pipe

{ - | u }
Toggles the set UID bit of the object, as follows:

- **u** Set UID on execution
- **-** Do not set UID on execution

{ - | g } Toggles the set group ID (GID) bit of the object, as follows:

- **g** Set GID on execution
- **-** Do not set GID on execution

{ - | t } Toggles the sticky bit of the object, as follows:

- **t** Sticky bit on
- **-** Sticky bit off

This 4-character token is required for every object.

Mode
Represents a string of 3 octal characters defining the read, write, and execute permissions of the object. The *Mode* token is required of every object. See the **chmod** command for more information about permissions.

Owner
Specifies the UID of the owner of the object. The owner token is required for every object.

Group
Specifies the GID of the owner of the object. The group token is required for every object.

Major Minor
Specifies the major and minor device numbers of the object if its type is a block or character special file. If the object is not a block or character special file, these tokens are omitted.

SourceFile	Applies only to regular file, hard link, and symbolic link objects. For regular files, this token is the path name to the file from which the object file is to be initialized. For both symbolic and hard links, this token is the source of the link. The source of the link is relative to the new file system for hard links.
DirectoryListing	Defines the contents of the object if it is a directory. The contents of the directory are defined using the token syntax described here. For example, a directory listing can include one or more regular files, one or more block files, and one or more directory listings. The **mkfs** command creates the directory entries . (dot) and .. (dot dot). Each directory listing is terminated with the special **$** token.
$	Ends the current directory listing or indicates the end of the prototype file.

Example Prototype Specification

The following prototype specification describes a JFS that does not have a boot program in block 0 and occupies the entire device. The 3rd token is ignored. The 4th and 5th tokens define the fragment size as 1024 bytes and the number of bytes per i-node as 2048. The main body of this prototype defines the file system contents.

```
<noboot> 0 0 1024 2048
d---  755 0 0
 dir1    d--- 755 0 2
  block_dev   b--- 644 0 0   880 881
  char_dev    c--- 644 0 0   990 991
  named_pipe  p--- 644 0 0
  regfile3    ---- 644 0 0   /tmp/proto.examp/dir1/regfile3
  regfile4    ---- 644 0 0   /tmp/proto.examp/dir1/regfile4
  $
 dir2    d--- 755 205 300
  regfile6   ---- 644 0 0    /tmp/proto.examp/dir2/regfile6
  symlnOutofFS l--- 644 0 0  /tmp/proto.examp/dir2/regfile6
  symlnNoExist l--- 644 0 0  /home/foobar
  symlnInFs  l--- 644 0 0    /dir2/regfile6
  regfile5   ---- 644 0 0    /tmp/proto.examp/dir2/regfile5
  hardlink   L--- 644 0 0    /dir2/regfile5
  $
 dir3   d--- 755 0 0
  setgid     --g- 755 0 0    /tmp/proto.examp/dir3/setgid
  setuid     -u-- 755 0 0    /tmp/proto.examp/dir3/setuid
  sticky     ---t 755 0 0    /tmp/proto.examp/dir3/sticky
  $
 dir4   d--- 755 0 0
  dir5  d--- 755 0 0
   dir6 d--- 755 0 0
    $
   dir7 d--- 755 0 0
    $
   $
  regfile7   ---- 644 0 0    /tmp/proto.examp/dir4/regfile7
  $
 regfile1    ---- 555 205 1  /tmp/proto.examp/regfile1
 regfile2    ---- 744 0 0    /tmp/proto.examp/regfile2
 $
$
```

Three entries for the dir2 object deserve further examination:

symlnOutofFS l-- 644 0 0 /tmp/proto.examp/dir2/regfile6	This entry defines a symbolic link to a file outside the file system to be created. The command ls -l lists something similar to symlnOutofFS -> /tmp/proto.examp/dir2/regfile6.
symlnNoExist l-- 644 0 0 /home/foobar	This entry defines a symbolic link to a file outside the file system to be created to a file that does not exist. The command ls -l lists something similar to symlnNoExist -> /home/foobar.
symlnInFs l-- 644 0 0 /dir2/regfile6	This entry defines a symbolic link to a file within the file system to be created. The command ls -l lists something similar to symlnInFS -> /dir/regfile6.

Examples

To make a prototype JFS using the prototype file described in the ″Example Prototype File Specification″ :

1. Generate the main body of the prototype file using the **proto** command or a text editor. For the purposes of this example, call the file /tmp/ProtoFile.
2. Add the first 5 tokens as required for a JFS. In the example prototype file, the tokens are:

   ```
   <noboot> 0 0 1024 2048
   ```
3. Create a logical volume to hold the file system, as follows:

   ```
   mklv -y protolv -t jfs SomeVGname 5
   ```

   This command creates a logical volume named protolv in the SomeVGname volume group. The size of the logical volume is 5 logical partitions.
4. Add an appropriate stanza to the /etc/filesystem file. A minimal example stanza is:

   ```
   /protofs:
     dev             = /dev/protolv
     vfs             = jfs
     log             = /dev/loglv00
     mount           = false
   ```
5. Run the following **mkproto** command:

   ```
   mkproto /dev/protolv /tmp/ProtoFile
   ```

   This command creates a JFS on the protolv logical volume. The size of the JFS is 5 logical partitions, its fragment size is 1024 bytes, and its nbpi ratio is 2048. The structure and contents of the file system are as specified in the prototype file /tmp/ProtoFile.

Files

/usr/sbin/mkproto	Contains the **mkproto** command.

Related Information

The **mkfs** command, **fsck** command, **fsdb** command, **proto** command.

The **filsys.h** file, **dir** file.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

mkprtldap Command

Purpose

Configures IBM Directory (LDAP) for Directory enabled System V print. It also configures client machines to use the Directory for System V print information.

Syntax

To configure the IBM Directory to store System V Print information:

mkprtldap -s -a *AdminDN* **-p** *Adminpasswd* **-w** *ACLBindPasswd* [**-f**] [**-d** *nodeDN*]

To configure clients to use the IBM Directory for System V Print information:

mkprtldap -c -h *DirectoryServerHostname* **-w** *ACLBindPasswd* [**-d** *PrintBindDN*] [**-U**]

To get usage information for the **mkprtldap** command:

mkprtldap ?

Description

The **mkprtldap** command configures the IBM Directory (LDAP) server, and one or more clients to use the Directory (LDAP) for System V Print information. This command must be run on the system being setup as the server and on all the client systems. Once the Directory (LDAP) server is configured for System V print, the directory enabled System V Print commands (**dslpadmin**, **dslpaccess**, **dslpsearch**, **dslpenable**, **dslpdisable**, **dslpaccept**, **lp**, **lpstat**, **cancel** and **dslpreject**) must be run to add , remove and manage System V print information (printers and print queues) on the Directory (LDAP) server. The **mkprtldap** command configures client machines to use the Directory (LDAP) server for System V print information.

The **mkprtldap** command requires the IBM Directory server software to be installed on the machine being configured as the server. The command also requires the IBM Directory client software to be installed on all client machines that will use the Directory (LDAP) server for System V print information.

Note: The client (**-c** flag) and the server (**-s** server) options cannot be run at the same time. When setting up a system as the server, the mkprtldap command should be run twice on that system. Once to set up the server, and again to set up the client.

During the server side configuration, using the **-s** flag, the **mkprtldap** command:

- Requires the IBM Directory Administrator's DN and password if the Directory has been configured. If the Directory Administrator's DN and password have not been set, **mkprtldap** will set them with the values passed to the command.
- Creates a db2 instance with **ldapdb2** as the default instance name.
- Creates a db2 database with **ldapdb2** as the default database name if one does no exist. If an existing database is found, **mkprtldap** adds AIX System V print information to the existing database.
- Creates the AIX Information tree DN (cn=aixdata container object) on the Directory if one is not present. The print subtree will be created under the AIX Information subtree. If an existing AIX Information subtree exists on the Directory, the print subtree will be created under it. All System V print information will be stored under the print subtree. The directory enabled System V print commands have to be run to add printers and print queues under the print subtree created.
- The default suffix and AIX Information tree for the **mkprtldap** command is a top level container object cn=aixdata. The Print subtree (ou=print) will be created under the AIX Information tree.

- The print subtree is ACL protected with the value of the **ACLBindPasswd** parameter passed to the command. The same value must be used when configuring clients to use the Directory for System V print information. Select a password value that is difficult for people or password cracking programs to guess.
- If the **-d** option is used and a valid existing node on the Directory is passed to the command, the AIX Information subtree is created under the given node. The print subtree is then created under the AIX Information subtree.
- Starts the IBM Directory server after all the above is done
- Adds the IBM Directory server process (slapd) to the **/etc/inittab** file to have the server start after a reboot.

During the client configuration, the **mkprtldap** command:

- Saves the IBM Directory (LDAP) server host name in the **/etc/ldapsvc/server.print** file.
- Saves the AIX Print Bind DN in the **/etc/ldapsvc/server.print** file.
- Saves the ACL Bind Password for the AIX Print Bind DN in the **/etc/ldapsvc/system.print** file. The value of the ACL Bind password must be the same as the one specified during the configuration of the Directory server.
- Undo a previous client configuration if the **-U** flag is specified. This option will replace the **/etc/ldapsvc/system.print** and **/etc/ldapsvc/server.print** files with the previous saved copies of the files (**/etc/ldapsvc/server.print.save** and **/etc/ldapsvc/system.print.save**).

Flags

Server

-a *AdminDN*	Specifies the Directory (LDAP) Administrator's DN.
-d *nodeDN*	This advanced option requires a valid existing node DN on the Directory under which the AIX Information tree and Print Subtree will be created.
-f	The force flag is required by the **mkprtldap** command to force the creation of the Print subtree (and AIX Information subtree if needed) when one or more AIX Information trees exist on the Directory.
-p *adminpasswd*	Specifies the Directory (LDAP) Administrator's password.
-s	Indicates the command is being run to configure the Directory for System V print.
-w *ACLBindPasswd*	Specifies the password to ACL protect the Print Subtree on the Directory. Select a password value that is difficult for people or password cracking programs to guess.

Client

-c	Indicates the command is being run to configure clients to use the Directory for System V Print information.
-d *PrintBindDN*	Specifies the Print Bind DN. The default Print Bind DN is **ou=print,cn=aixdata**. The Print Bind DN to use during Client configuration is displayed at the end of the server setup of the **mkprtldap** command.
-h *DirectoryServerHostname*	Hostname of the IBM Directory server setup to store System V Print information.
-U	Undo a previous configuration of a client.
-w *ACLBindPasswd*	The ACL Bind Password for the print subtree. The ACL Bind password is specified during the server setup of the **mkprtldap** command. The value of the ACL Bind Password must match the one used during the setup of the Directory server.

Usage

? Displays usage information for the **mkprtldap** command.

Security

This command can be run by the `root` user only.

Examples

1. To configure a new installation of IBM Directory for System V print with the Administrator DN **cn=root** and password root, type:

   ```
   mkprtldap -s -a cn=root -p root -w aclpasswd
   ```

   where the ACL Bind password is the password used to ACL protect the print subtree. The ACL Bind password is specified during the configuration of System V Print on the Directory. This configuration will also set the Directory Administrator's DN and password to **cn=root** and **root**. Running the command will setup a suffix and top level object **cn=aixdata**. The Print subtree (**ou=print**) will be created under this AIX Information tree (**cn=aixdata object**). Select a ACL Bind password value that is difficult for people or password cracking programs to guess.

2. To configure System V print on a machine with a configured IBM Directory server -

   The Administrator DN and password are required to configure System V print on the Directory. Assume the existing Administrator's DN and password are **cn=admin** and **passwd**.

   ```
   mkprtldap -s -a cn=admin -p passwd -w pass123wd
   ```

3. The **mkprtldap** command provides the option to configure the IBM Directory to store the print information under a pre-existing node (e.g. **o=ibm**,**c=us**) on the Directory [Advanced Option]. This is only recommended when it is necessary to store the print information under the existing node on the Directory for specific reasons. The recommend option is to store the print subtree in the default location on the Directory by not specifying the **-d** option. The Administrator DN and password are required to configure System V print on the Directory Assume the existing Administrator's DN and password are **cn=admin** and **passwd**.

   ```
   mkprtldap -a cn=admin -p passwd -w acl123passwd -d o=ibm,c=us
   ```

   Running the command will create an AIX Information tree (**cn=aixdata**) under the **o=ibm,c=us** object. The print subtree will be created under this new object (**cn=aixdata**, **o=ibm**, **c=us**).

4. To configure System V print on a machine with a configured IBM Directory server and an existing AIX Information tree. There might be situations where the Directory contains an existing AIX information tree with other subsystem specific information (e.g Security or NIS information). It might be required to store the print information in a separate location on the Directory under a different AIX Information tree. The command, by default, will not create a new AIX Information tree if one exists on the Directory. To force the command to create a new AIX Information tree to store the print information, use the **-f** flag with the command. Consider the case where the Security and NIS subsystem information is stored under the AIX Information tree at **cn=aixdata,o=ibm,c=us**. To create a new AIX Information tree for print information different from the existing one, run the command with the **-f** flag and specify the default location or another node. The Administrator DN and password are required to configure System V print on the Directory. Assume the existing Administrator's DN and password are **cn=admin** and **passwd**

   ```
   mkprtldap -a cn=admin -p passwd -w passwd123 -f
   ```

   Running the command will create a new AIX Information tree (**cn=aixdata**) with the suffix (**cn=aixdata**) and the print information will be stored under this new AIX Information tree (**ou=print**, **cn=aixdata**). There will be two AIX Information trees on the Directory in this example **cn=aixdata,o=ibm,c=us** and **cn=aixdata**. The print information will be under the **cn=aixdata** object (suffix - **cn=aixdata**). For **mkprtldap**, it is recommend to use the default location to add the print information to the Directory.

5. To configure a client to use an IBM Directory setup for System V Print on host **server.ibm.com**, type:

   ```
   mkprtldap -c -h server.ibm.com -w passwd
   ```

   Please ensure that the ACL Bind Password (**passwd**) is the same as the one specified during the setup of the Directory Server. Running the command without specifying a Print Bind DN value with the -d option will cause the command to use the default Print Bind DN **ou=print,cn=aixdata**. The Print Bind DN must match the one displayed at the end of running the **mkprtldap** command to configure the server.
6. To change the information in the client side configuration files, run the **mkprtldap** command with the new information

   ```
   mkprtldap -c -h server.ibm.co.uk -w aclpasswd -d ou=print,cn=aixdata,c=uk
   ```

   Executing this command on a client that has already been configured will change the information in the **/etc/ldapsvc/server.print** and **/etc/ldapsvc/system.print** files to contain the new configuration information. The original contents of the **/etc/ldapsvc/server.print** and **/etc/ldapsvc/system.print** will stored in the **/etc/ldapsvc/server.print.save** and **/etc/ldapsvc/system.print.save** files.

Files

Mode	File	Description
rw	**/etc/slapd32.conf**	(Server configuration) - Contains the IBM Directory (LDAP) configuration information.
rw	**/etc/ldapsvc/server.print**	(Client configuration) - Contains information about the Directory Server configured to store System V Print information. (Machine name, Location of Print subtree on the Directory and LDAP port)
rw	**/etc/ldapsvc/system.print**	(Client configuration) - Contains the ACL Bind Password for the Print subtree on the Directory.

Related Information

Configuring Directory Enabled System V print in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

The **dslpadmin** command, **dslpaccess** command, **dslpsearch** command, **dslpenable** command, **dslpdisable** command, **dslpaccept** command, **lp** command, **lpstat** command, **cancel** command, **dslpreject** command, and **mksecldap** command.

The **/etc/ldapsvc/server.print** and **/etc/ldapsvc/system.print** files.

The **/etc/slapd32.conf** file.

---

mkprtsv Command

Purpose

Configures TCP/IP-based print service on a host.

Syntax

To Configure and Start Print Service for a Client Machine

mkprtsv -c [**-S**][**-q** *QueueName* **-v** *DeviceName* **-b** *"Attribute=Value ..."* **-a** *"Attribute=Value ..."* | **-A** *FileName*]

To Configure and Start Print Service for a Server Machine

mkprtsv -s [**-S**] [**-q** *QueueName* **-v** *DeviceName* **-b** *"Attribute=Value ..."* **-a** *"Attribute=Value ..."* | **-A** *FileName*] [**-h** *"HostName ..."* | **-H** *FileName*]

Description

The **mkprtsv** high-level command configures a TCP/IP-based print service on a host. The print service configuration can be done for a host functioning as a client or for a host functioning as a server.

Use the command to configure and start the print service.

To configure print service for a client, the **mkprtsv** command calls the spooler **mkque** and **mkquedev** commands to change the **/etc/lpd/qconfig** file (or its object class equivalent) appropriately and set up a spooler queue on the client machine.

To configure print service for a server, the **mkprtsv** command does the following:

1. Calls the **ruser** command to set up remote users to print on the server.
2. Calls the **mkque** and **mkquedev** commands to change the server's **/etc/lpd/qconfig** file appropriately and set up the necessary device queues on the server machine.
3. Calls the **startsrc** command to activate the **lpd** and **qdaemon** server daemons. The **qdaemon** server daemon starts the **piobe** printer backend.

Flags

-A *FileName*
: Specifies name of file containing entries related to the **qconfig** file.

-a *"Attribute* **=** *Value..."*
: Specifies a list of attributes and their corresponding values to be used for updating the spooler's **qconfig** file or object class. The **-a** flag is optional. Valid attribute types are listed below:

  acctfile (true/false)
  : Identifies the file used to save **print** command accounting information. The default value of **false** suppresses accounting. If the named file does not exist, no accounting is done.

  argname
  : Specifies the logical printer name.

  device
  : Identifies the symbolic name that refers to the device stanza.

  discipline
  : Defines the queue-serving algorithm. The default value of **fcfs** means first come, first served. A **sjn** value means shortest job next.

  pserver
  : Specifies the remote print server.

  up (true/false)
  : Defines the state of the queue. The default value of **true** indicates that it is running. A **false** value indicates that it is not.

-b *"Attribute =Value..."* Specifies a list of attributes and their corresponding values to be used for updating the spooler's **qconfig** file or object class. At least one attribute must be defined for the **-b** option. The **backend** attribute is required. Valid attribute types are listed below:

access (true/false)
Specifies the type of access the backend has to the file specified by the **file** attribute. The **access** attribute has a value of **write** if the backend has write access to the file, or a value of **both** if the backend has both read and write access. This field is ignored if the file field has a value of **false**.

align (true/false)
Specifies whether the backend sends a form-feed control before starting the job if the printer has been idle. The default value is **false**.

backend
Specifies the full path name of the backend, optionally followed by flags and parameters to be passed to it. The **backend** attribute is required.

feed Specifies the number of separator pages to print when the device becomes idle, or takes a **never** value, which indicates that the backend is not to print separator pages.

file Identifies the special file where the output of the backend is to be redirected. The default value of **false** indicates no redirection. In this case, the backend opens the output file.

header (never/always/group)
Specifies whether a header page prints before each job or group of jobs. The default value of **never** indicates no header page. To produce a header page before each job, specify an **always** value. To produce a header before each group of jobs for the same user, specify a **group** value.

trailer (never/always/group)
Specifies whether a trailer page prints after each job or group of jobs. The default value of **never** indicates no trailer page. To produce a trailer page after each job, specify an **always** value. To produce a trailer after each group of jobs for the same user, specify a **group** value.

host Specifies the host name from which to print.

s_statfilter
Translates short queue-status information to a format recognized by this operating system.

l_statfilter
Translates long queue-status information to a format recognized by this operating system.

-c Performs print service configuration for a client machine. Use the **-q** flag with the **-c** option.

-H *FileName* Specifies the name of a file containing a list of host names.

-h *"HostName..."* Specifies a list of host names to be included in the list of remote users who can use the print server. The queuing system does not support multibyte host names.

-q *QueueName* Specifies the name of a queue in the **qconfig** file.

-S Starts print service after it is configured. If the **-S** flag is omitted, print service is configured but not started.

-s Performs print service configuration for a server machine. Use the **-h**, **-H**, and **-q** flags with the **-s** flag.

-v *DeviceName* Specifies the name of the device stanza in the **qconfig** file.

Examples

1. To configure and enable print service for a client, enter the command in the following format:

```
mkprtsv  -c  -S  -a"argname=rp1 backend=piobe \
pserver=print802"
```

In this example, `rp1` is the logical printer name, `piobe` is the printer backend, and `print802` is the remote print server.

2. To configure a print server using initialization information and allow remote printing, enter the command in the following format:

```
mkprtsv  -s  -H hnames  -A qinfo
```

In this example, attribute information stored in the `qinfo` file initializes the spooler, and the list of host names stored in the `hnames` file is the list of remote hosts that have access rights to the print server.

Files

/etc/lpd/qconfig	Contains configuration information for the printer queuing system.

Related Information

The **mkque** command, **mkquedev** command, **qadm** command, **ruser** command, **startsrc** command.

The **lpd** daemon, **qdaemon** daemon.

TCP/IP Daemons in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

mkps Command

Purpose

Adds an additional paging space.

Syntax

To Add a Logical Volume for Additional Paging Space

mkps [**-a**] [**-n**] [**-t** *lv*] **-s** *LogicalPartitions VolumeGroup* [*PhysicalVolume*]

To Add Additional Paging Space On an NFS Server

mkps [**-a**] [**-n**] **-t** *nfs ServerHostName ServerFileName*

Description

The **mkps** command adds an additional paging space. Before the paging space can be used it must be activated, using the **swapon** command. The *VolumeGroup* parameter specifies the volume group within which the logical volume for the paging space is to be made. The *PhysicalVolume* parameter specifies the physical volume of the *VolumeGroup* on which the logical volume is to be made.

In the second form of the **mkps** command, the *ServerHostName* parameter specifies the NFS server where the *ServerFileName* resides. The *ServerFileName* specifies the file which will be used for the NFS paging of the system. The *ServerFileName* file must exist and be exported correctly to the client that will use the file for paging.

You can use the File Systems application in Web-based System Manager (wsm) to change file system characteristics. You could also use the System Management Interface

Flags

-a	Specifies that the paging space is configured at subsequent restarts.
-n	Activates the paging space immediately.
-s *LogicalPartitions*	Specifies the size of the paging space and the logical volume to be made in logical partitions.
-t	Specifies the type of paging space to be created. One of the following variables is required:
	lv Specifies that a paging space of type logical volume should be created on the system.
	nfs Specifies that a paging space of type NFS should be created on the system.

Examples

1. To create a paging space in volume group `myvg` that has four logical partitions and is activated immediately and at all subsequent system restarts, enter:

   ```
   mkps  -a  -n  -s4 myvg
   ```

2. To create an NFS paging space on the NFS server `swapserve` where the host `swapserve` has the **/export/swap/swapclient** file exported, enter:

   ```
   mkps -t nfs swapserve /export/swap/swapclient
   ```

Files

/etc/swapspaces	Specifies the paging space devices that are activated by the **swapon -a** command.

Related Information

The **chps** command, **lsps** command, **rmps** command, **mklv** command, **swapon** command.

Paging Space Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

File Systems in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The System Management Interface Tool (SMIT) in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkqos Command

Purpose

Configures the system to support QoS.

Syntax

/usr/sbin/mkqos [**-I** | **-N** | **-B**]

Description

The **mkqos** command configures the system to support Quality of Service (QoS).

Flags

-B	Adds an entry to the **inittab** file to execute the **/etc/rc.qos** file now and on the next system restart. This flag is the default.
-I	Adds an entry to the **inttab** file to execute the **/etc/rc.qos** file on the next system restart.
-N	Executes the **/etc/rc.qos** file to start the QoS daemons. When invoked in this way, the QoS daemons run until the next system restart.

Files

inittab	Controls the initialization process of the system.
/etc/rc.qos	Contains the startup script for the QoS daemons.

Related Information

The **rmqos** command.

QoS Installation and **TCP/IP Quality of Service (QoS)** in the *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

mkque Command

Purpose

Adds a printer queue to the system.

Syntax

mkque [**-D**] **-q** *Name* [**-a** '*Attribute* **=** *Value*' ...]

Description

The **mkque** command adds a printer queue to the system by adding the stanza described on the command line to the end of the **/etc/qconfig** file.

You can use the Printer Queues application in Web-based System Manager (wsm) to change printer characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkque** fast path to run this command.

To use the SMIT fast path to go directly to the **Add a Local Queue** dialog, enter:

```
smit mklque
```

To use the SMIT fast path to go directly to the **Add a Remote Queue** dialog, enter:

```
smit mkrque
```

> **Note:** Do not edit the **/etc/qconfig** file while there are active jobs in any queue. Editing includes both manual editing and use of the **chque**, **mkque**, **rmque**, **mkquedev**, **rmquedev**, or **chquedev** commands. It is recommended that all changes to the **/etc/qconfig** file be made using these commands. However, if manual editing is desired, first issue the **enq -G** command to bring the

queuing system and the **qdaemon** daemon to a halt after all jobs are processed. Then edit the **/etc/qconfig** file and restart the **qdaemon** daemon with the new configuration.

Flags

-a *'Attribute* **=** *Value'*
: Specifies a line to be added to the queue stanza in the **/etc/qconfig** file. This flag must be the last flag when entering the **mkque** command on the command line. For a list of all valid attributes, see the **/etc/qconfig** file.

  Note: It is recommended that you do not use the `'device = '` attribute. This attribute is handled automatically by the **mkquedev** command. Also note that the queuing system does not support multibyte host names.

-D
: Specifies that the queue defined by the *Name* variable queue is added to the top of the **/etc/qconfig** file and is therefore the default queue. If you do not specify this flag, the *Name* variable is added to the bottom of the **/etc/qconfig** file and is not the default queue.

-q *Name*
: Specifies the name of the queue to be added.

  Note: The queue name must not exceed 20 characters.

Examples

To add the print queue `lp0` specifying a host name of `leo` and a remote print queue named `lp013`, enter:

```
mkque  -qlp0  -a 'host = leo' -a 'rq = lp013'
```

Files

/usr/bin/mkque	Contains the **mkque** command.
/etc/qconfig	Configuration file.

Related Information

The **chque** command, **lsque** command, **mkquedev** command, **rmque** command.

The **qconfig** file.

Printer Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Specific Information in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Printer Support in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

---

mkquedev Command

Purpose

Adds a printer queue device to the system.

Syntax

mkquedev -d *Name* **-q** *Name* **-a** '*Attribute* **=** *Value*' ...

Description

The **mkquedev** command adds a printer queue device to the system by adding the stanza described on the command line to the **/etc/qconfig** file.

You can use the Printer Queues application in Web-based System Manager (wsm) to change printer characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkquedev** fast path to run this command.

> **Note:** Do not edit the **/etc/qconfig** file while there are active jobs in any queue. Editing includes both manual editing and use of the **chque**, **mkque**, **rmque**, **mkquedev**, **rmquedev**, or **chquedev** commands. It is recommended that all changes to the **/etc/qconfig** file be made using these commands. However, if manual editing is desired, first issue the **enq -G** command to bring the queuing system and the qdaemon to a halt after all jobs are processed. Then edit the **/etc/qconfig** file and restart the qdaemon with the new configuration.

Flags

-a '*Attribute* **=** *Value*'
: Specifies the '*Attribute* **=** *Value*' attribute to be added to the device stanza in the **/etc/qconfig** file. This flag must be the last flag when entering the **mkquedev** command on the command line. For a list of valid attributes, see the **/etc/qconfig** file.

  Note: The **'backend = '** attribute must be included when entering this command on the command line.

-d *Name*
: Specifies with the *Name* variable the name of the queue device to add.

  Note: The queue device name must not exceed 20 characters.

-q *Name*
: Specifies with the *Name* variable the name of the queue (this name must already exist) to which the queue device is added. The **mkquedev** command automatically adds the **'device = '** attribute to the specified queue stanza.

Examples

To add the `postscript` print queue device to the `lp0` queue, specify the backend program to be the **piobe** command (`backend = /usr/lib/lpd/piobe`) and direct the backend program not to align the paper (`align = FALSE`), enter:

```
mkquedev  -qlp0  -dpostscript  -a 'backend = /usr/lib/lpd/piobe' \
 -a 'align = FALSE'
```

Files

/usr/bin/mkquedev	Contains the **mkquedev** command.
/etc/qconfig	Contains the configuration file.

Related Information

The **chquedev** command, **lsquedev** command, **mkque** command, **rmquedev** command.

The **/etc/qconfig** file.

Printer Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Adding a Print Queue Device in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Support in the *AIX 5L Version 5.2 Guide to Printers and Printing*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

mkramdisk Command

Purpose

Creates a RAM disk using a portion of RAM that is accessed through normal reads and writes.

Syntax

mkramdisk [**-u**] *size*[**M** | **G**]

Description

The **mkramdisk** command is shipped as part of **bos.rte.filesystems**, which allows the user to create a RAM disk. Upon successful execution of the **mkramdisk** command, a new RAM disk is created, a new entry added to **/dev**, the name of the new RAM disk is written to standard output, and the command exits with a value of 0. If the creation of the RAM disk fails, the command prints an internalized error message, and the command will exit with a nonzero value.

The size can be specified in terms of MB or GB. By default, it is in 512 byte blocks. Suffix *M* will be used to specify size in megabytes and *G* to specify size in gigabytes.

The names of the RAM disks are in the form of **/dev/rramdisk***x* where x is the logical RAM disk number (0 through 63).

The **mkramdisk** command also creates block special device entries (for example, **/dev/ramdisk5**) although use of the block device interface is discouraged because it adds overhead. The device special files in **/dev** are owned by root with a mode of 600. However, the mode, owner, and group ID can be changed using normal system commands.

Up to 64 RAM disks can be created.

Note: The size of a RAM disk cannot be changed after it is created.

The **mkramdisk** command is responsible for generating a major number, loading the ram disk kernel extension, configuring the kernel extension, creating a ram disk, and creating the device special files in **/dev**. Once the device special files are created, they can be used just like any other device special files through normal **open**, **read**, **write**, and **close** system calls.

RAM disks can be removed by using the **rmramdisk** command. RAM disks are also removed when the machine is rebooted.

To set up a RAM disk that is approximately 20 MB in size and create a file system on that RAM disk, enter the following:

```
mkramdisk 40000
ls -l /dev | grep ram
mkfs -V jfs /dev/ramdiskx
mkdir /ramdisk0
mount -V jfs -o nointegrity /dev/ramdiskx /ramdiskx
```

where *x* is the logical RAM disk number.

By default, RAM disk pages are pinned. Use the **-u** flag to create RAM disk pages that are not pinned.

Flags

-u Specifies that the ram disk that is created will not be pinned. By default, the ram disk will be pinned.

Parameters

size Indicates the amount of RAM (in 512 byte increments) to use for the new RAM disk. For example, typing:

```
mkramdisk 1
```

creates a RAM disk that uses 512 bytes of RAM. To create a RAM disk that uses approximately 20 MB of RAM, type:

```
mkramdisk 40000
```

Exit Status

The following exit values are returned:

0 Successful completion.
>0 An error occurred.

Examples

1. To create a new ram disk using a default 512-byte block size, and the size is 500 MBs (1048576 * 512), enter:

   ```
   $ mkramdisk 1048576
     /dev/rramdisk0
   ```

   The **/dev/rramdisk0** ramdisk is created.
2. To create a new ramdisk with a size of 500 Megabytes, enter:

   ```
   $ mkramdisk 500M
     /dev/rramdisk0
   ```

   The **/dev/rramdisk0** ramdisk is created. Note that the ramdisk has the same size as example 1 above.
3. To create a new ram disk with a 2-Gigabyte size, enter:

   ```
   $ mkramdisk 2G
     /dev/rramdisk0
   ```

Files

/usr/sbin/mkramdisk	Contains the **mkramdisk** command.

Related Information

The **rmramdisk** command.

---

mkresponse Command

Purpose

Creates a new response definition.

Syntax

To create a response with no actions:

mkresponse [**–p** *node_name*] [**–h**] [**–TV**] *response*

To create a response with one action:

mkresponse –n *action* [**–d** *days_of_week*[**,***days_of_week...*]]
[**–t** *time_of_day*[**,***time_of_day...*]] **–s** *action_script* [**–r** *return_code*]
[**–e a** | **r** | **b**] [**–o**] [**–E** *env_var*=*value*[**,***env_var*=*value*...]]
[**–u**] [**–p** *node_name*] [**–h**] [**–TV**] *response*

To copy a response:

mkresponse –c *existing_response*[**:***node_name*] [**–p** *node_name*] [**–h**] [**–TV**] *response*

Description

The **mkresponse** command creates a new response definition with the name specified by the *response* parameter. One action can also be specified when the response is defined. Actions define commands to be run when the response is used with a condition and the condition occurs. The action defines days of the week when the action can be used, the time of day for those days of the week, the script or command to be run, what type of event causes the command to be run, the expected return code of the script or command, and whether to keep standard output. The days and times are paired so that different times can be specified for different days. A response with no actions only logs the events.

When in a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the response definition. If the **-p** flag is not specified, the response is defined on the local node.

The **chresponse** command can be used to add actions to a response or to remove actions from a response. Monitoring can be started by using the **startcondresp** command. The **startcondresp** command links a response to a condition if they are not already linked.

Flags

–c *existing_response***[:***node_name***]**
: Copies an existing response. Links with conditions are not copied. The existing response is defined on the node known as *node_name* in a domain. 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.

–n *action*
: Specifies the name of the action being defined. Only one action can be defined when the response is created. Use the **chresponse** command to add more actions to the response.

–d *days_of_week*

Specifies the days of the week when he action being defined can be run. *days_of_week* and *time_of_day* together define the interval when the action can be run.

Enter the numbers of the days separated by a plus sign (+) or as a range of days separated by a hyphen (-). More than one *days_of_week* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default is all days. If no value is specified but a comma is entered, the default value is used. The numbers of the days follow:

1 Sunday
2 Monday
3 Tuesday
4 Wednesday
5 Thursday
6 Friday
7 Saturday

–t *time_of_day*

Specifies the time range when *action* can be run, consisting of the start time followed by the end time, separated by a hyphen. *days_of_week* and *time_of_day* together define the interval when the action can be run.

The time is in 24–hour format (HHMM) where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. More than one *time_of_day* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default value is 0000-2400. If no value is specified but a comma is entered, the default value is used.

–s *action_script*

Specifies the fully-qualified path for the script or command to run for the action being defined. See the man pages for **logevent**, **notifyevent**, and **wallevent** for descriptions of the predefined response scripts provided with the application.

–r *return_code*

Specifies the expected return code for *action_script*. If the expected return code is specified, the actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating whether they match. If the **–r** flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

–e a | **r** | **b**

Specifies the type of event that causes the action being defined to run:

a Event. This is the default.

r Rearm event.

b Both event and rearm event.

–o Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

–E *env_var*=*value*[,*env_var*=*value*...]

Specifies any environment variables to be set before running the action. If multiple *env_var*=*value* variables are specified, they must be separated by commas.

–u Specifies that the action is to be run when a monitored resource becomes undefined.

–p *node_name*

Specifies the name of the node where the response is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

–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

response The *response* name is a character string that identifies the response. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the **IBM.EventResponse** resource class to run **mkresponse**. 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.

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 define a response with the name ″Log event in audit log″, run this command:

   ```
   mkresponse "Log event in audit log"
   ```

2. To define a response with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command:

   ```
    mkresponse -n "E-mail root" -d 1+7  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
   "E-mail root any time"
   ```

3. To define a response with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used anytime Saturday and Sunday but only 8 am to 5 pm Monday through Friday and that uses the command **/usr/sbin/rsct/bin/notifyevent root** for events, run this command::

   ```
   mkresponse -n "E-mail root"     \
   -d 1+7,2-6  -t 0000-2400,0800-1700  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e a  \
   "E-mail root any time"
   ```

4. To define a response with the name ″E-mail root any time″ that has an action named ″E-mail root″ to be used any time Saturday and Sunday, that uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, and that sets the environment variable LANG to en_US, run this command:

   ```
   mkresponse -n "E-mail root" -d 1+7  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e b \
   -E LANG="en_US"  "E-mail root any time"
   ```

5. To define a response with the name ″E-mail root first shift″ that has an action named ″E-mail root″ to be used Monday through Friday from 8 am to 6 pm, that uses the command **/usr/sbin/rsct/bin/notifyevent root** for rearm events, and that saves standard output in the audit log, expecting return code 5, run this command:

   ```
   mkresponse -n "E-mail root" -d 2-6  -t 0800-1800  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e r  -o   \
   -r 5  "E-mail root first shift"
   ```

6. To define a response with the name ″Critical notifications″ as a copy of ″Warning notifications″, enter:

   ```
   mkresponse -c "Warning notifications" "Critical notifications"
   ```

These examples apply to management domains:

1. To define a response on the management server with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

   ```
    mkresponse -n "E-mail root" -d 1+7  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
   "E-mail root any time"
   ```

2. To define a response on the managed node **nodeB** with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

   ```
    mkresponse -n "E-mail root" -d 1+7  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
   -p nodeB "E-mail root any time"
   ```

3. To define a response on the managed node **nodeB** with the name ″nodeB Warning notifications″ as a copy of ″nodeA Warning notifications″ on the managed node **nodeA**, run this command on the management server:

```
mkresponse -c "nodeA Warning notifications":nodeA \
-p nodeB "nodeB Warning notifications"
```

These examples apply to peer domains:

1. To define a response on the current node with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
"E-mail root any time"
```

2. To define a response on the node **nodeB** in the domain with the name ″E-mail root any time″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday, that uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, and that sets two environment variables (PAGE ALL and TIMER SET), run this command from any node in the domain:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
-p nodeB -E 'ENV1="PAGE ALL", ENV2="TIMER SET"' \
"E-mail root any time"
```

3. To define a response on the node **nodeB** in the domain with the name ″nodeB Warning notifications″ as a copy of ″nodeA Warning notifications″ on the node **nodeA** in the domain, run this command from any node in the domain:

```
mkresponse -c "nodeA Warning notifications":nodeA \
-p nodeB "nodeB Warning notifications"
```

Location

/usr/sbin/rsct/bin/mkresponse
Contains the **mkresponse** command

Related Information

ERRM commands: **chresponse**, **lsresponse**, **mkcondition**, **mkcondresp**, **rmresponse**, **startcondresp**

Files: **rmccli**

Books: see the *RSCT Administration Guide* for more information about ERRM operations

---

mkrole Command

Purpose

Creates new roles. This command applies only to AIX 4.2.1 and later.

Syntax

mkrole [*Attribute*=*Value* ...] *Name*

Description

The **mkrole** command creates a new role. The *Name* parameter must be a unique role name. You cannot use the **ALL** or **default** keywords as the role name.

You can use the Users application in Web-based System Manager to change user characteristics. You could also use the System Management Interface Tool (SMIT) to run this command.

Restrictions on Creating Role Names

To prevent inconsistencies, restrict role names to characters with the POSIX portable filename character set. You cannot use the keywords **ALL** or **default** as a role name. Additionally, do not use any of the following characters within a role-name string:

: Colon
″ Double quotation mark
#
, Comma
= Equal sign
\ Backslash
/ Slash
? Question mark
’ Single quotation mark
` Back quotation mark

Finally, the *Name* parameter cannot contain any space, tab, or new-line characters.

Restrictions on Creating Roles

To ensure the integrity of the role information, only users with the **RoleAdmin** authorization can create a role.

Parameters

Attribute=*Value*	Initializes a role attribute. Refer to the **chrole** command for the valid attributes and values.

Security

Files Accessed:

Mode	File
rw	/etc/security/roles
r	/etc/security/user.roles

Auditing Events:

Event	Information
ROLE_Create	role

Examples

To create the `ManageObjects` role and set the **groups** attribute to `objects`, type:

```
mkrole groups=objects ManageObjects
```

Files

/etc/security/roles	Contains the attributes of roles.
/etc/security/user.roles	Contains the role attribute of users.

Related Information

The **chrole** command, **chuser** command, **lsrole** command, **lsuser** command, **mkuser** command, **rmrole** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

Administrative Roles Overview in *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

mkrpdomain Command

Purpose

Creates a new peer domain definition.

Syntax

mkrpdomain [**–t** *TS_port*] [**–g** *GS_port*] [**–c**] [**–h**] [**–TV**] *domain_name node_name1* [*node_name2* ...]

mkrpdomain [**–t** *TS_port*] [**–g** *GS_port*] [**–c**] **–f** *file_name* [**–h**] [**–TV**]

Description

The **mkrpdomain** command creates a new peer domain definition with the name specified by the *domain_name* parameter. The nodes specified by *node_name* are defined to the new peer domain. A peer domain can be used to provide high-availability services when configuring application and system resources.

The **preprpnode** command must have been run on each of the nodes to be defined to the peer domain. The **preprpnode** command prepares the security environment for the peer domain operations. See the **preprpnode** command for more information about peer domain definition requirements. Only those nodes that have the appropriate security setup will be successfully defined to the peer domain.

If the UDP port numbers for group services and topology services are not available on all of the nodes to be defined to the peer domain, the **mkrpdomain** command will fail. The command will also fail if the domain name is already being used or if any node cannot be successfully defined to the peer domain.

The **mkrpdomain** command does not bring the peer domain online automatically. To bring the peer domain online, run the **startrpdomain** command. Nodes can be added to and removed from the peer domain using the **addrpnode** and **rmrpnode** commands.

A node can be defined in more than one peer domain but it can be online in only one peer domain at a time.

Flags

-t *TS_port*
: Specifies the topology services port number. This UDP port is used for daemon-to-daemon communication. Any unused port in the range 1024 to 65535 can be assigned. The command will fail if the specified port is unavailable. The default is 12347.

-g *GS_port*
: Specifies the group services port number. This UDP port is for daemon-to-daemon communication.

Any unused port in the range 1024 to 65535 can be assigned. The command will fail if the specified port is unavailable. The default is 12348.

–c Continues to run the **mkrpdomain** command on the remaining nodes.

By default, if the **mkrpdomain** command fails on any node, it will fail on all nodes. The **-c** flag overrides this behavior, so that the **mkrpdomain** command will run on the other nodes, even if it fails on one node.

–f *file_name*
Reads a list of node names from *file_name*.

-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

domain_name Specifies the name of the new peer domain to be created. The name can contain any printable character, except it cannot contain the slash character (**/**). In addition, the name cannot be **IW**.

node_name1 **[***node_name2* **...]**
Specifies the node (or nodes) to include in this peer domain definition. The node name is the IP address or the long or short version of the DNS hostname. The node name must resolve to an IP address.

Security

The user of the **mkrpdomain** command needs write permission to the **IBM.PeerDomain** resource class on each node that is to be defined to the peer domain. This is set up by running the **preprpnode** command on each node that is to be defined to the domain, specifying the name of the node on which the user will run **mkrpdomain**.

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

Any node to be defined to the peer domain must be reachable from the node on which this command runs.

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

1. To define a peer domain consisting of **nodeA** with the domain name **ApplDomain**, run one of these commands on **nodeA**:

   ```
   mkrpdomain ApplDomain

   mkrpdomain ApplDomain nodeA
   ```

2. To define a peer domain consisting of **nodeA**, **nodeB**, and **nodeC** with the domain name **ApplDomain**, run this command on **nodeA**, **nodeB**, or **nodeC**:

   ```
   mkrpdomain ApplDomain nodeA nodeB nodeC
   ```

3. To define a peer domain consisting of **nodeA** and **nodeB** with domain name **ApplDomain**, topology services port number 1200, and group services port number 2400, run this command on **nodeA** or **nodeB**:

   ```
   mkrpdomain -t 1200 -g 2400 ApplDomain nodeA nodeB
   ```

Location

/usr/sbin/rsct/bin/mkrpdomain
Contains the **mkrpdomain** command

Files

The **/etc/services** file is modified.

Related Information

Commands: **addrpnode**, **lsrpdomain**, **lsrpnode**, **preprpnode**, **rmrpdomain**, **rmrpnode**, **startrpdomain**, **stoprpdomain**

---

mkrset Command

Purpose

Makes an rset containing the specified CPUs and memory regions and places it in the system registry.

Syntax

```
mkrset -c CPUlist [ -m MEMlist ] rsetname
```

Description

The **mkrset** command creates and places into the system registry an **rset** with the specified set of CPUs and/or memory regions. The rset name must not exist in the registry. The owner and group IDs of the rset will be set to the owner and group IDs of the command issuer. The rset will have read/write owner permissions and read permission for group and other.

Flags

-c List of CPUs to be in the rset. This can be one or more CPUs or CPU ranges.
-m List of memory regions to be in the rset. This can be one or more memory regions or ranges.

Parameters

rsetname The name of the rset to be placed in the system registry. The name consists of a *namespace* and an *rsname* separated by a "/" (slash). Both the namespace and *rsname* may contain up to 255 characters. See the **rs_registername()** service for additional information about character set limits of rset names.

Security

The user must have `root` authority or CAP_NUMA_ATTACH capability.

Examples

1. To make an **rset** containing CPUs 0-7 named **test/cpus0to7**, type:
   ```
   mkrset -c 0-7 test/cpus0to7
   ```
2. To make an **rset** containing CPUs 1, 3, 5, 6, 7, 10 named **test/lotsofcpus**, type:
   ```
   mkrset -c 1 3 5-7 10 test/lotsofcpus
   ```

Files

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

Related Information

The **attachrset**, **detachrset**, **execrset**, **lsrset**, and **rmrset** commands.

---

mkrsrc Command

Purpose

Defines a new resource.

Syntax

To define a new resource using data that is entered on the command line:

mkrsrc [**–v**] [**–h**] [**–TV**] *resource_class attr*=*value...*

To define a new resource using data that is predefined in an input file:

mkrsrc –f *resource_data_input_file* [**–v**] [**–h**] [**–TV**] *resource_class*

To see examples of the **mksrc** command for a resource class:

mkrsrc –e [**–h**] [**–TV**] *resource_class*

Description

The **mkrsrc** command requests that the RMC subsystem define a new resource instance for the class specified by the *resource_class* parameter. At least one persistent attribute name and its value must be specified either as a parameter or by a resource definition file using the **-f** flag.

Before you run **mkrsrc**, you should run the **lsrsrcdef** command to determine which attributes are designated as ″required for define″ or ″optional for define″. Only attributes that are designated as ″required for define″ or ″optional for define″ can be defined using the **mkrsrc** command. The **lsrsrcdef** command also identifies the data type for each attribute. The value specified for each attribute must match this data type.

Flags

–e Displays two examples of suitable command-line input for this command:

1. An example of **mkrsrc** command-line input for required attributes only.
2. An example of **mkrsrc** command line input for both required and optional attributes.

–f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information.

–v Verifies that the input file or the command line is valid. For example, it will make sure all of the attribute names specified are defined as persistent attributes and are designated as ″required for define″ or ″optional for define″.

–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

resource_class
Specifies the resource class name of the resource to be defined.

attr=*value...*
Specifies the attributes of the resource being defined. When defining a new resource instance, there are specific required attributes for each resource that must be defined. These attributes can be specified as parameters on the command line or defined in an input file by using the **-f** flag.

attr The name of a persistent attribute for this resource. This attribute must be designated as ″required for define″ or ″optional for define″. Use the **lsrsrcdef** command to check the designation.

value The value for this persistent attribute. The data type for this value must match the defined data type for the value of this attribute. Use the **lsrsrcdef** command to verify the data type for each attribute.

Security

The user needs write permission for the *resource_class* specified in **mkrsrc** to run **mkrsrc**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0 Command has run successfully.

1 Error occurred with RMC.

2 Error occurred with CLI script.

3 Incorrect flag on command line.

4 Incorrect parameter on command line.

5 Error occurred with RMC that was based on faulty command line input.

Environment Variables

CT_CONTACT

When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. 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.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To create a new resource in the **IBM.Host** class, assuming you already know which persistent attributes are required when defining a resource of this class, enter:

   ```
   mkrsrc IBM.Host Name=c175n05
   ```

2. To create a new resource in the **IBM.Processor** class by first generating a template to aid in the defining of these resources, enter:

   ```
   lsrsrcdef -i IBM.Processor > /tmp/IBM.Processor.rdef
   ```

   Then, edit the file **/tmp/IBM.Processor.rdef** and enter values for all of the attributes, substituting the type for an appropriate value, or leaving it blank for the default value.

   Finally, enter:

   ```
   mkrsrc -f /tmp/IBM.Processor.rdef IBM.Processor
   ```

3. To create two new IBM.Host resources using the information defined in file **/tmp/IBM.Host.rdef**, enter:

   ```
   mkrsrc -f /tmp/IBM.Host.rdef IBM.Host
   ```

where the file **/tmp/IBM.Host.rdef** looks like this:

```
PersistentResourceAttributes::
resource 1:
    Name        = c175n04

resource 2:
    Name        = c175n05
```

4. This example creates a new resource in the IBM.Foo class. In this class, **Name** and **NodeList** are required attributes. The Binary, SD, StringArray, and SDArray attributes are optional. This example shows how to enter the more difficult data types from the command line. The data types for the optional Binary, SD, StringArray, and SDArray attributes are self-explanatory. Enter:

```
mkrsrc IBM.Foo Name=c175n05 \
NodeList={1} \
Binary="0xaabbccddeeff00" \
SD='[testing123,1,{2,4,6}]' \
StringArray='{"testing 1 2 3",testing123,"testing 1 2 3"}' \
SDArray='{["testing 1 2 3",1,{1,3,5}],[testing,2,{2,4,6}]}'
```

   Note: As discussed in the **rmccli** general information file, attribute values for certain data types (structured data, array of structured data, and arrays containing strings enclosed in double quotation marks) should be enclosed in single quotation marks.

Location

/usr/sbin/rsct/bin/mkrsrc Contains the **mkrsrc** command

Related Information

Commands: **lsrsrc**, **lsrsrcdef**, **mkrsrc**

Files: **Resource_Data_Input**, **rmccli**

Books: see the *RSCT Administration Guide* for information about RMC operations

---

mkseckrb5 Command

Purpose

Migrates existing operating system users to Kerberos.

Syntax

mkseckrb5 [**-h** | **-r**]

Description

This command gets the list of user names and creates Kerberos users. If the **-r** flag is not specified, the command prompts for a new password for each user.

Standard Output Consists of information messages when the **-h** flag is used.
Standard Error Consists of error messages when the command cannot complete successfully.

Flags

-h Specifies that the command is only to display the valid command syntax.

-r Specifies that random passwords are to be used.

Exit Status

Failure of this command to execute successfully results in incomplete migration. The admin must check the Kerberos database for the users that were migrated before taking further action.

0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

Only the root user is authorized to use this command.

Examples

1. To display the command syntax, type:

   ```
   mkseckrb5 -h
   ```

2. To migrate existing users to Kerberos users, type:

   ```
   mkseckrb5
   ```

3. To migrate user trojan to Kerberos user with random passwd, type:

   ```
   mkseckrb5 -r trojan
   ```

Files

/usr/sbin/mkseckrb5	Contains the **mkseckrb5** command.

---

mksecldap Command

Purpose

Sets up an AIX cluster to use LDAP for security authentication and data management.

Syntax

The syntax to set up a server is:

mksecldap -s -a *adminDN* **-p** *adminpasswd* **-S** *schematype* [**-d** *baseDN*] [**-n** *port*] [**-k** *SSLkeypath*] [**-w** *SSLkeypasswd*] [**-u** *NONE*] [**-U**]

The syntax to set up a client is:

mksecldap -c -h *serverlist* **-a** *adminDN* **-p** *adminpasswd* [**-d** *baseDN*] [**-n** *serverport*] [**-k** *SSLkeypath*] [**-w** *SSLkeypasswd*] [**-t** *cachetimeout*] [**-C** *cachesize*] [**-P** *NumberofThreads*] [**-T** *heartBeatInt*] [**-u** *userlist*] [**-U**]

Description

The **mksecldap** command can be used to set up IBM SecureWay Directory servers and clients for security authentication and data management. This command must be run on the server and all clients.

Notes:

1. The client (**-c** flag) and the server (**-s** flag) options cannot be run at the same time. When setting up a server, the **mksecldap** command should be run twice on that machine. Once to set up the server, and again to set up the client.
2. The SecureWay Directory server configuration file is **/etc/slapd.conf** for SecureWay Directory 3.1 or earlier and **/etc/slapd32.conf** for SecureWay Directory 3.2 or later. AIX 5.2 only supports SecureWay Directory 3.2 and later.

For server setup, make sure that the **ldap.server** fileset is installed. When installing the **ldap.server** fileset, the **ldap.client** fileset and the backend DB2 software are automatically installed as well. No DB2 preconfiguration is required to run this command for LDAP server setup. When you run the **mksecldap** command to set up the server, the command will:

1. Create a DB2 instance with **ldapdb2** as the default instance name.
2. Create a DB2 database with **ldapdb2** as the default database name. If a database already exists, **mksecldap** will bypass the above two steps. (This is the case when the LDAP server has been set up for other usage.) The **mksecldap** command will use the existing database to store the AIX user/group data.
3. Create the AIX tree DN (suffix). If no baseDN is supplied from the command line, the default suffix is set to **cn=aixdata** and the user/group data is placed to the **cn=aixsecdb,cn=aixdata** DN. This is the recommended case. Otherwise, the **mksecldap** command takes the user-supplied DN and prefixes it with **cn=aixdata** and makes the newly constructed DN the suffix. This behavior is shown in the following table. The value within the brackets represents the optional user supplied DN from command line.

CMD-line DN:	[o=ibm]
suffix:	cn=aixdata[,o=ibm]
security DN:	cn=aixsecdb,cn=aixdata[,o=ibm]
user DN:	ou=aixuser,cn=aixsecdb,cn=aixdata[,o=ibm]
group DN:	ou=aixgroup,cn=aixsecdb,cn=aixdata[,o=ibm]

   In case the LDAP server has already been setup in the local system, the **mksecldap** command searches for the **cn=aixsecdb** keyword from the suffixes defined in the **slapd32.conf** configuration file and from the database. If it finds the keyword, it assumes that **mksecldap** has been run, and bypasses the base DN setup step and the user/group migration step, and exits.

   If **cn=aixsecdb** is not found in the suffixes and the database, the **mksecldap** command checks for the **cn=aixdata** keyword. **cn=aixdata** is a common base DN shared by various AIX LDAP components. If the **mksecldap** command find the keyword, it compares it with the user supplied DN. If they are the same, the users/groups will be put under the **cn=aixsecdb,cn=aixdata,[userDN]**. If they are different, the **mksecldap** command prints an error message warning the existence of the **cn=aixdata,...** DN, and it will not migrate the users/groups under the user supplied DN. You can choose to use the existing **cn=aixdata,...** by running the **mksecldap** command again with that existing DN.
4. Migrates the data from the security database files from the local host into the LDAP database. Depending on the **-S** option, the **mksecldap** command migrates users/groups using one of the three LDAP schemas:
   - **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** object classes).

     Attention: Systems running AIX 4.3 and AIX 5.1 which are configured as LDAP clients will only work with servers of AIX schema type. They will not talk to ldap servers of RFC2307 or RFC2307AIX types.
5. Set the LDAP server administrator DN and password. This name/password pair is also used for access control of the AIX tree.
6. Set the SSL (secure socket layer) for secure data transfer between this server and the clients. This setup requires the **GSKIT** to be installed.

   Note: If this option is used, the SSL key must be created before running the **mksecldap** command. Otherwise the server may not be able to start.

7. Installs the **/usr/ccs/lib/libsecldapaudit.a**, a LDAP server plug-in. This plugin supports AIX audit of the LDAP server.
8. Start/restart the LDAP server after all the above is done.
9. Add the LDAP server process (**slapd**) to **/etc/inittab** to have the LDAP server start after reboot.
10. With the **-U** option, undo a previous setup for the server configuration file. The first time you run the **mksecldap** command, it saves two copies of the **slapd32.conf** server configuration file. One is saved to **/etc/security/ldap/slap32.conf.save.orig** and the other to **/etc/ security/ldap/slapd32.conf.save**. Each subsequent run of **mksecldap**, the **current slapd32.conf** is only saved to **/etc/security/ldap/slapd32.conf.save** file. The undo option restores the **/etc/slapd32.conf** server configuration file with the **/etc/security/ ldap/slapd32.conf.save** copy.

    Note: The undo option applies to the server configuration file only. It has no effect on the database.

Note: All the LDAP configuration is saved into the **/etc/slapd32.conf** LDAP server configuration file.
For client setup, make sure the LDAP server has been setup and is running. The **mksecldap** command does the follows during for client setup:

1. Saves the LDAP server(s)' host name.
2. Saves the user base DN and group base DN of the server. If no **-d** option is supplied from command line, the **mksecldap** command searches the LDAP server for **aixaccount**, **aixaccessgroup**, **posixaccount**, **posixgroup**, and **aixauxaccount** objectclasses from the LDAP server, and set up the base DNs accordingly. If the server has multiple user/group bases, you must supply the **-d** option with a RDN so that the **mksecldap** command can setup the base DNs to the ones within that RDN.

    If the **posixaccount** objectclass is found during client setup, **mksecldap** will also try to search for base DNs for these entities: hosts, networks, services, netgroups, protocols, and rpc from the server and save any that is found.
3. Determines the schema type used by the LDAP server - **AIX** specific schema, **RFC 2307** schema, or **RFC 2307** schema with full AIX support (see objectclasses listed in step 2). It sets the objectclasses and attribute maps in the **/etc/security/ldap/ ldap.cfg** file accordingly. The **mksecldap** command does not recognize other schema types, so clients must be setup manually.
4. Sets SSL for secure data transfer between this host and the LDAP server. This step requires that the client SSL key and the key password are created in advance, and the server must be setup to use SSL for the client SSL to work.
5. Saves the LDAP server administrator DN and password. The DN/password pair must be the same as the pair specified during server setup.
6. Sets the cache size in terms of the number of entries used by the client side daemon. Valid values are 100-10,000 for users and 10-1,000 for groups. The default value is 1,000 for users and 100 for groups.
7. Sets the cache timeout of the client side daemon. Valid values are 60-3600 seconds. The default is 300 seconds. Set this value to 0 to disable caching.
8. Sets the number of threads used by the client side daemon. Valid values are 1-1,000. The default is 10.
9. Sets the time interval in seconds that the client daemon checks for the LDAP server status. Valid value are 60-3,600 seconds. The default is 300.
10. Optionally sets the list of users or all users to use LDAP by modifying their SYSTEM line in the **/etc/security/user** file. For more information on enabling ldap login in, see the following note.
11. Starts the client daemon process (**secldapclntd**).
12. Adds the client side daemon process to **/etc/inittab** to have this daemon start after a reboot.
13. With the **-U** option, undo a previous setup to the **/etc/security/ldap/ldap.cfg** file.

Note: The client configuration data is saved to the **/etc/security/ldap/ldap.cfg** file. Setting the SYSTEM to `LDAP` for the default stanza of **/etc/security/user** only allows LDAP users to login to the system. Setting the SYSTEM to `LDAP` or `compat` allows both LDAP users and local users to login to the system.

Flags

For Server Setup

-a *AdminDN*	Specifies the LDAP server administrator DN.
-d *baseDN*	Specifies the suffix or base DN of the AIX subtree. The default is **cn=aixdata**.
-k *SSLkeypath*	Specifies the full path to the SSL key database of the server.
-n *port*	Specifes the port number that the LDAP server listens to. Default is 389 for non-SSL and 636 for SSL.
-p *adminpasswd*	Specifies the clear text password for the administrator DN.
-S *schematype*	Specifies the LDAP schema used to represent user/group entries in the LDAP server. Valid values are AIX, RFC2307, and RFC2307AIX.
-s	Indicates that the command is being run to setup the server.
-w *SSLkeypasswd*	Specifies the password for the SSL key.
-U	Specifies to undo the previous server setup to the LDAP configuration file.
-u `NONE`	Specifies not to migrate users and groups from local system. The only valid value is `NONE`. Any other values are ignored. No **-S** option is required with this option.

For Client Setup

-a *AdminDN*	Specifies the LDAP server administrator DN. It must match the one used for the server setup.
-c	Indicates the command is being run to setup the client.
-C *Cachsize*	Specifies the maximum number of user entries used in the client side daemon cache. Valid values are 100-10,000 for user cache. The default value is 1,000. The group cache is 10% of that of user cache.
-d *baseDN*	Specifies the base DN for the **mksecldap** command to search for the user base DN and group base DN. If not specified from the command line, the entire database is searched.
-h *serverlist*	Specifies a comma separated list of hostnames (server and backup servers).
-k *SSLkeypath*	Specifies the full path to the client SSL key.
-n *serverport*	Specifies the port number that the LDAP server is listening to.
-p *adminpasswd*	Specifies the clear text password for the administrator DN of the LDAP server. It must match the one used for the server setup.
-P *NumberofTreads*	Specifies the number of threads the client side daemon uses. Valid values are 1-1,000. The default is 10.
-t *Cachetimeout*	Specifies the maximum time length that a cache entry expires. Valid val-ues are 60-3,600 seconds. The default is 300 seconds. Set this value to 0 to disable caching.
-T *heartBeatInt*	Specifies the time interval of heartbeat between this client and the LDAP server. Valid values are 60-3,600 seconds. Default is 300.
-u *userlist*	Specifies the comma separated list of usernames. Specify **ALL** to enable all users on the client.
-w *SSLkeyfilepath*	Specifies the password for the client SSL key.
-U	Specifies to undo the previous client setup to the LDAP client configura-tion file.

Examples

1. To setup a LDAP server of AIX specific schema for users and groups, enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -S aix
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is migrated from local files to the default **cn=aixdata** suffix.

2. To setup a LDAP server with a baseDN other than the default and with SSL secure communication , enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -d o=mycompany,c=us -S rfc2307 \ -k /usr/ldap/serverkey.kdb
    -w keypwd
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is migrated from local files to the **cn=aix-data,o=mycompany,c=us** suffix. The LDAP server uses SSL communications by using the key stored at **/usr/ldap/serverkey.kdb**. The password to the key, **keypwd**, must also be supplied. Users and groups are migrated with the RFC 2307 schema.

3. To undo a previous server setup:

   ```
   mksecldap -s -U
   ```

   This undo the previous setup to the **/etc/slapd32.conf** server configuration file. Note, for safty reasons, this does not remove any database entries or database created my a previous setup. One has to remove the database entries/database manually if they are not needed any more.

4. To setup a client to use the **server1.ibm.com** and **server2.ibm.com** LDAP servers, enter:

   ```
   mksecldap -c -a cn=admin -p adminpwd -h server1.ibm.com,server2.ibm.com
   ```

   The LDAP server adminstrator DN and password mut be supplied for this client to authenticate to the server. The **mksecldap** command contacts the LDAP server for schema type used, and sets up the client accordingly. Without the **-d** option from the command line, the entire server DIT is searched for the user base DN and the group base DN.

5. To setup the client to talk to the **server3.ibm.com** LDAP server using SSL, enter:

   ```
   mksecldap -c -a cn=admin -p adminpwd -h server3.ibm.com -d o=mycompany,c=us -k /usr/ldap/clientkey.kdb -w keypwd -u user1,user2
   ```

   This sets up a LDAP client similar to case 3, but with SSL communication. The **mksecldap** command searches the **o=mycompany,c=us** RDN for user base DN and group base DN. Account user1 and user2 are configured to authenticate through LDAP.

   Note: The **-u ALL** option enables all LDAP users to login to this client.

6. To undo a previous client setup, enter:

   ```
   mksecldap -c -U
   ```

   This undo the previous setup to the **/etc/security/ldap/ldap.cfg** file. This does not remove the **SYSTEM=LDAP** and **registry=LDAP** from the **/etc/security/user** file.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/group**
r	**/etc/security/passwd**
r	**/etc/security/limits**
r	**/etc/security/user (on the server)**
rw	**/etc/security/user (on the clients)**
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/roles**
rw	**/etc/security/login.cfg (on the server)**
rw	**/etc/slapd32.conf (on the server)**
rw	**/etc/security/ldap/ldap.cfg (on the client)**

Related Information

The **secldapclntd** daemon.

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

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

LDAP Exploitation of the Security Subsystem in *AIX 5L Version 5.2 Security Guide*.

---

mksecpki Command

Purpose

mksecpki configures AIX PKI server components. The components of AIX PKI are Certificate Authority, Registration Authority, and Audit subsystems.

Syntax

mksecpki {**-u** *username* **-f** *reference_file* [**-p** *CA_port*] [**-H** *ldap_host*] [**-D** *dn* **-w** *password*] [**-i** *certificate_issuer_dn*] | **-U** *username*}

Description

The **mksecpki** command configures AIX PKI server components. **mksecpki** must be run after configuring an LDAP server to publish certificates. The values for the options **-H**, **-D**, **-w**, and **-i** must be the same values as the ones specified during the LDAP configuration. Otherwise, the CA will not be able to publish certificates to LDAP.

The **-u** option specifies the AIX username which will host AIX PKI. The username must follow AIX username rules. Do not use **-u** and **-U** together. The invoker of the command will be asked to provide a password for the username. **mksecpki** will create a database instance with the same name.

The **-f** option specifies the file containing the reference number and passphrase. The client certificate requests will use these exact same values while communication with the CA. The reference number and passphrase are each specified on a separate line. The following is the contents of an example **iafile**:

```
11122233
temppwd1234
```

The **-p** option specifies the port that Certificate Authority accepts the certificate requests. If no port number is given, 1077 will be assumed.

The **-H** option specifies the hostname of the LDAP server where the certificates are published to. Prior to invoking the **mksecpki** command, an LDAP server must be setup to publish certificates. Otherwise, the

certificates will not be published to LDAP, however, certificate will be returned to the requestor when certificate management commands are used. If the **-H** option is not given the localhost will be used as the hostname.

The **-D** option is used to specify the directory administrators distinguished name. This must be the same one that is specified during the configuration of the LDAP server.

The **-w** option specifies the password corresponding to the administrator DN. It is an error not to specify both the admin DN and password.

The **-i** option specifies the distinguish name of the Certificate Authority issuing the certificates. This must be the same value as the one given when setting an LDAP server for publishing certificates.

The **-U** option specifies the username that hosts the AIX PKI that will be unconfigured. The command will confirm the unconfiguration before starting its operation. This option removes the username from the system. The invokers of this command will be asked if they want to remove the home directory of the username. When this command runs without errors, it displays a message indicating the successful completion. The invoker of this command is recommended to wait for this message.

Flags

-u *username* Specifies the name of the username that is going to be created that will host AIX PKI server components.

-f *reference_file* Specifies the file which contains the reference number and passphrase that is used when making a certificate creation request.

-p *CA_port* Specifies the Certificate Authority Communication Port.

-H *ldap_host* Specifies the LDAP host where the certificates are going to be published.

-D *adminDN* Specifies directory administrator distinguished name (DN).
Note: The **-D** option requires that the **-w** *password* option also be specified.

-w *password* Specifies directory administrator password.

-i *certificate_issuer_dn* Specifies the distinguished name of the Certificate Authority issuing certificates.

-U *username* Specifies the username which hosts the AIX PKI that will be unconfigured.

Security

This command should grant execute (x) access only to the `root` user and members of the security group.

Examples

To configure AIX PKI server side using **pkitest.ibm.com** as the LDAP host name for publish certificates and using o=aix,c=us as the issuer name, enter:

```
$ mksecpki -u pkiuser -f iafile -p 829 -H pkitest.ibm.com -D cn=admin
-w password -i o=aix,c=us
```

where **iafile** contains the reference number and passphrase.

To unconfigure the server, enter:

```
$ mksecpki -U pkiuser
```

Files

/usr/lib/security/pki/ca.cfg

Related Information

The **certadd**, **certcreate**, **certdelete**, **certget**, **certlink**, **certlist**, **certrevoke**, **certverify**, **keyadd**, **keydelete**, **keylist**, and **keypasswd** commands.

---

mksensor Command

Purpose

Adds an event sensor command to the resource monitoring and control (RMC) subsystem.

Syntax

mksensor [**–i** *seconds*] [**–n** *host*] [**–h**] [**–v** | **–V**] *name command*

Description

The **mksensor** command adds the specified command to the resource monitoring and control (RMC) subsystem to be run periodically to retrieve a user-defined value from the system. Usually this value is used to cause an event to occur by means of the event response resource manager (ERRM). For example, if the specified command returns the number of users logged on to the system (a variable that none of the standard RMC resource managers return currently), an ERRM condition and response can be written to run a response when the number of users logged on exceeds a certain value. This enables administrators to extend the RMC monitoring capabilities without having to write a resource manager.

The specified command returns the value it retrieves from the system by sending it to standard output in the form *attr*=*value*. The attribute name used depends on the type of the value and is one of these: **String**, **Int32**, **Uint32**, **Int64**, **Uint64**, **Float32**, **Float64**, or **Quantum**. If only the value is sent to standard output, the attribute name is assumed to be **String**.

If more than one type of data is to be returned, a series of *attr*=*value* pairs are sent to standard output, separated by blanks, for example: **Int32=10 String=**″**abcdefg**″. These attribute names are also the dynamic attributes that can be used in the ERRM condition. The exit value of the script is stored in the dynamic attribute **ExitValue** and can also be used in an ERRM condition. When the ERRM condition is written, the **IBM.Sensor** resource class should be specified, and the selection string should normally be *name*=’*name*’, where *name* is the name of the sensor being defined by this **mksensor** command.

The *command* parameter represents the command that should be run periodically to retrieve the desired system value. By default, this command is run by RMC under the same user name that originally ran the **mksensor** command to add this sensor.

The **mksensor** command can be run on the management server or on any node. It normally operates on the local node where it is run unless the CT_CONTACT environment variable is set. If you want **mksensor** to run on all of the nodes in a domain, use the CT_MANAGEMENT_SCOPE environment variable. See the **Environment variables** section for more information.

This command returns the error messages and exit status of the underlying RMC command **mkrsrc**.

Flags

–i *seconds*
: Specifies how often to periodically run the command. This interval is in seconds. The minimum value is 10 seconds. The default value is 60 seconds.

–n *host*
: Specifies the node on which the sensor should be created. By default, the sensor is created on the local node. This flag is only appropriate in a management domain or a peer domain.

–h Writes the command's usage statement to standard output.

–v | –V
Writes the command's verbose messages to standard output. Use this flag to see the underlying RMC command that is being run. This can help to interpret error messages that are displayed.

Parameters

name Specifies the name of the sensor that is being created.

command
Specifies the command that should be run to retrieve the system value that this sensor monitors.

Security

The user needs write permission for the **IBM.Sensor** resource class in order to run **mksensor**. 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 has run successfully.

1 An incorrect combination of flags and parameters has been entered.

If an error occurs in the RMC layer, the RMC return code will be returned as the exit status.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_MANAGEMENT_SCOPE
Determines the management scope used for the command's RMC daemon session.

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.

Examples

1. To create a new sensor called **NumLogins** that runs a script called **numlogins** to find out how many users are currently logged in, enter:

   ```
   mksensor NumLogins /usr/local/bin/numlogins
   ```

   The **numlogins** script looks like this:

   ```
   #!/usr/bin/perl
   my @output=`who`;
   print 'Int32='scalar(@output), "\n";
   exit;
   ```

Location

/usr/sbin/rsct/bin/mksensor Contains the **mksensor** command

Related Information

Commands: **chsensor**, **lscondition**, **lssensor**, **mkcondition**, **mkcondresp**, **mkresponse**, **mkrsrc**, **rmsensor**

Files: see the **rmccli** man page for information about *attr*=*value* syntax.

Books: see the *RSCT Administration Guide* for information about the ACL authorization file, ERRM, and ERRM environment variables.

---

mkserver Command

Purpose

Adds a subserver definition to the subserver object class.

Syntax

mkserver -c *CodePoint* **-s** *Subsystem* **-t** *Type*

Description

The **mkserver** command adds a subserver definition to the **Subserver** object class.

Flags

-c *CodePoint*	Specifies the *CodePoint* integer that identifies the subserver. This is the value by which the subsystem knows the subserver. The **mkserver** command is unsuccessful if this *CodePoint* value already exists for this subsystem. The limit for *CodePoint* storage is the same as a short integer (1 through 32,768).
-s *Subsystem*	Specifies the name that uniquely identifies the subsystem to which the subserver belongs. The **mkserver** command is unsuccessful if the *Subsystem* name is not known in the subsystem object class, or if the *Subsystem* name is that of a known subsystem in the subsystem object class but uses signals as its communication method.
-t *Type*	Specifies the name that uniquely identifies the subserver. The **mkserver** command is unsuccessful if the *Type* name is already known in the **Subserver Type** object class.

Security

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **mkserver** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Addserver	Lists in an audit log subsystems that have been added and the entire Object Data Management record.

See ″Setting up Auditing″ in *AIX 5L Version 5.2 Security Guide* for more details about how to properly select and group audit events and how to configure audit event data collection.

Examples

To add a subserver definition, enter:

```
mkserver  -s srctest  -t tester  -c 1234
```

This adds a subserver definition to the **Subserver Type** object class, with an owning subsystem of `srctest` and a subserver code point of `1234`.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.

Related Information

The **auditpr** command, **chserver** command, **rmserver** command, **startsrc** command, **stopsrc** command.

System Resource Controller Overview in the *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Auditing Overview in the *AIX 5L Version 5.2 Security Guide*.

System Resource Controller (SRC) Overview for Programmers in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Defining Your Subsystem to the SRC in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

---

mkslave Command

Purpose

Invokes the **ypinit** command to retrieve maps from an NIS master server and starts the **ypserv** daemon to configure a slave server.

Syntax

/usr/sbin/mkslave [**-C** | **-c**] [**-O** | **-o**] [**-I** | **-B** | **-N**] *Master*

Description

The **mkslave** command invokes the **ypinit** command to retrieve maps from the master server you specify on the command line. The **ypserv** daemon starts after the **ypinit** command has completed successfully. Use the *Master* parameter to specify the host name of the master server. The master server specified should already be configured and running.

You can use the Network application in Web-based System Manager (wsm) to change network characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkslave** fast path to run this command.

Flags

-C Invokes the **ypinit** command with the **-n** flag. The **mkslave** command continues on errors. This flag is the default.
-c Stops execution when errors occur.
-O Overwrites any maps that exist in the domain.
-o Prevents the overwriting of maps that exist in the domain. This flag is the default.
-I Invokes the **ypinit** command immediately but does not start the **ypserv** daemon until the next system reboot.
-N Invokes the **ypinit** command and starts the **ypserv** daemon.
-B Invokes the **ypinit** command, starts the **ypserv** daemon and configures the **ypserv** to start at system reboot. This flag is the default.

Examples

To invoke the **ypinit** command so that the master server `host2` will be contacted for maps, enter:

```
mkslave -O host42
```

This command will overwrite the current maps.

Files

/var/yp/*DomainName* directory	Contains the NIS maps for the NIS domain.

Related Information

The **chmaster** command, **mkclient** command, **rmyp** command, **smit** command, **ypinit** command.

The **ypbind** daemon, **yppasswdd** daemon, **ypserv** daemon, **ypupdated** daemon.

Network Information Services (NIS) Overview for System Management in *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT) Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

NIS Reference.

---

mkssys Command

Purpose

Adds a subsystem definition to the subsystem object class.

Syntax

mkssys { **-p** *Path* **-s** *Subsystem* **-u** *UserID* } [**-a** *Arguments*] [**-e** *StandardError*] [**-i** *StandardInput*] [**-o** *StandardOutput*] [**-t** *Synonym*] [**-O** | **-R**] [**-d** | **-D**] [**-q** | **-Q**] [**-K** | [**-I** *MessageQueue* **-m** *MessageMType* | **-f** *StopForce* **-n** *StopNormal* **-S**]] [**-E** *Nice*] [**-G** *Group*] [**-w** *Wait*]

Description

The **mkssys** command adds a new subsystem definition to the subsystem object class. If no flags are chosen after the **-p**, **-s**, and **-u** flags have been specified, the defaults are **-e /dev/console**, **-i /dev/console**, **-o /dev/console**, **-O**, **-d**, **-Q**, **-K**, **-E 20**, and **-w 20**.

> **Note:** Any auditing performed by the System Resource Controller (SRC) when actions are taken for the subsystem is logged against the login ID of the user who created the subsystem by using the **mkssys** command. For example, if you are logged in with root user authority, the subsystem is added with root user authority as the audit account.

Flags

-a *Arguments*	Specifies any arguments that must be passed to the command, started as the subsystem. These *Arguments* variables are passed by the SRC to the subsystem according to the same rules used by the shell. For example, quoted strings are passed as a single argument, and blanks outside a quoted string delimit arguments. Single and double quotes can be used.
-d	Specifies that inactive subsystems are displayed when the **lssrc -a** command (status all) request is made. By default, if the **-D** and **-d** flags are not present, the **-d** flag is used.
-D	Specifies that inactive subsystems are not displayed when status-all or status-group requests are made.
-e *StandardError*	Specifies where the subsystem *StandardError* data is placed. If the **-e** flag is not specified, the **/dev/console** file is used for standard error.
-E *Nice*	Changes the execution priority of the subsystem. Valid values are 0 through 39 (ordinary *Nice* variables are mapped to all positive numbers). If the **-E** flag is not present, the subsystem priority defaults to 20. Values between 0 and 19 are reserved for users with root authority.
-f *StopForce*	Specifies the signal sent to the subsystem when a forced stop of the subsystem is requested. Use only when the subsystem uses signals. The **mkssys** command is unsuccessful if the *StopForce* parameter is not a valid signal.
-G *Group*	Specifies that the subsystem belongs to the *Group* specified, and that the subsystem responds to all group actions on the *Group*.
-i *StandardInput*	Specifies where the subsystem standard input is routed. This field is ignored when the subsystem uses sockets communication. If the **-i** flag is not specified, by default the **/dev/console** file is used for standard input.
-I *MessageQueue*	Specifies that the subsystem uses message queues as the communication method. The *MessageQueue* variable specifies the message queue key for creating the message queue for the subsystem. Use the **ftok** subroutine with the subsystem path name as input to generate a unique key.
-K	Specifies that the subsystem uses sockets as its communication method. If a communication method is not specified, sockets communication is used by default.
-m *MessageMType*	Specifies the message type key the subsystem expects on packets sent to the subsystem by the SRC. Use only when the subsystem uses message queues communication.
-n *StopNormal*	Specifies the signal sent to the subsystem when a normal stop of the subsystem is requested. Use only when the subsystem uses signals communication. The **mkssys** command is unsuccessful if the *StopNormal* variable is not a valid signal.
-o *StandardOutput*	Specifies where the subsystem standard output is placed. If the **-o** flag is not specified, by default the **/dev/console** file is used for standard out.
-O	Specifies that the subsystem is not restarted if it stops abnormally. The default is no restart.
-p *Path*	Specifies the absolute path to the subsystem executable program.
-q	Specifies that the subsystem can have multiple instances running at the same time.
-Q	Specifies that multiple instances of the subsystem are not allowed to run at the same time and the subsystem is not to share the same interprocess communication (IPC) queue. If the **-q** flag is not specified, the **-Q** flag is the default.
-R	Specifies that the subsystem is restarted if the subsystem stops abnormally.
-s *Subsystem*	Specifies a name that uniquely identifies the subsystem. The **mkssys** command is unsuccessful if the subsystem name is already known in the subsystem object class.
-S	Specifies that the subsystem uses the signals communication method. You cannot define subservers for a subsystem name when your communication method is signals.
-t *Synonym*	Specifies an alternate name for the subsystem. The **mkssys** command is unsuccessful if the synonym name is already known in the subsystem object class.
-u *UserID*	Specifies the user ID for the subsystem. The *UserID* that creates the subsystem is used for security auditing of that subsystem.

-w *Wait* Specifies the time, in seconds, allowed to elapse between a stop cancel (**SIGTERM**) signal and a subsequent **SIGKILL** signal. Also used as the time limit for restart actions. If the subsystem stops abnormally more than twice in the time limit specified by the *Wait* value, the subsystem is not automatically restarted. By default, if the **-w** flag is not present, the wait time default is 20 seconds.

Security

Auditing Events: If the auditing subsystem has been properly configured and is enabled, the **mkssys** command will generate the following audit record (event) every time the command is executed:

Event	Information
SRC_Addssys	Lists in an audit log the name of the subsystem being added to the Object Data Manager (ODM) database and the entire ODM record.

See ″Setting up Auditing″ in *AIX 5L Version 5.2 Security Guide* for details about selecting and grouping audit events, and configuring audit event data collection.

Examples

1. To add a subsystem that uses sockets as its communication type, type the following:

   ```
   mkssys  -s srctest  -p /usr/lpp/srctest/srctest  -u 0  -K
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of sockets, a user ID of `0` (root), and a subsystem name of `srctest`.
2. To add a subsystem that uses message queues as its communication type, type the following:

   ```
   mkssys  -s srctest  -p /usr/lpp/srctest/srctest  -u 0  -I 123456 \ >  -m 789
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of message queues, a message queue key of `123456`, and a subsystem message type of `789`.
3. To add a subsystem that uses signals as its communication type, type:

   ```
   mkssys  -s srctest  -p /usr/lpp/srctest/srctest  -u 0  -S  -n 30 \ >  -f 31
   ```

   This adds a subsystem definition to the subsystem object class, with a communication type of signals, a stop normal signal of `30`, a stop force signal of `31`.
4. To add a subsystem that uses sockets as its communication type and is always passed an argument, type:

   ```
   mkssys  -s srctest  -p /usr/lpp/srctest/srctest  -u 0  -a ″-x″
   ```

   This adds a subsystem definition to the subsystem object class with a communication type of sockets and a command argument of `″-x″`.

Files

/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/dev/SRC	Specifies the **AF_UNIX** domain in the **socket.h** file.
/dev/.SRC-unix	Specifies the location for temporary file sockets.

Related Information

The **auditpr** command, **chssys** command, **lssrc** command, **refresh** command, **rmssys** command, **startsrc** command, **stopsrc** command, **traceson** command, **tracesoff** command.

System Resource Controller Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Auditing Overview in *AIX 5L Version 5.2 Security Guide*.

System Resource Controller (SRC) Overview for Programmers in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

Defining Your Subsystem to the SRC in *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs*.

---

mkstr Command

Purpose

Creates an error message file.

Syntax

mkstr [**-**] *MessageFile Prefix File* ...

Description

The **mkstr** command creates a file of error messages that can be removed from a single C source file or from multiple source files. Its use can reduce the size of programs that contain many error diagnostics and reduce system overhead in running such programs, because error messages are then not constantly swapped in and out of the source files.

The **mkstr** command processes each file specified by the *File* parameter, placing a massaged version of the file in a file having the name specified by the *Prefix* parameter followed by the original name.

To process the error messages in the source to the file specified by the *MessageFile* parameter, the **mkstr** command keys on the string **`error(**"**'** in the input stream. The string, starting at the **'**"**'** (two double quotation marks), is placed in the message file and followed by a null character and a new-line character. The null character terminates the message so it can be easily used when retrieved. The new-line character makes it possible to see the contents of the error message file by using the **cat** command.

The massaged copy of the input file then contains an **lseek** pointer into the file, which can be used to retrieve the message to its appropriate source file, as shown in the following example:

```
char efilname[] = "/usr/lib/pistrings";
int    efil = -1;

error(a1, a2, a3, a4)
{
    char buf[256];
    if (efil < 0) {
        efil = open(efilname, 0);
        if (efil < 0) {
oops:
           perror(efilname);
            exit(1);
        }
     }
    if (lseek(efil,(long) a1,0) <0||
```

```
        read(efil,buf, 256) <= 0)
       goto oops;
    printf(buf, a2, a3, a4);
}
```

Flags

- The optional - (minus sign) causes the error messages to be placed at the end of the *MessageFile* for recompiling part of a large mkstr program.

Examples

1. To put the error messages from the current directory C source files into the file `pistrings` and to put processed copies of the source for these files into file names prefixed by `xx`, enter*:*

   ```
   mkstr pistrings xx *.c
   ```

2. To append the error messages from an additional source file into the file `pistrings`, enter:

   ```
   mkstr - pistrings xx newfile.c
   ```

Files

/usr/ccs/bin/mkstr	Contains the **mkstr** command.

Related Information

The **cat** command, **xstr** command.

The **lseek** subroutine.

---

mksysb Command

Purpose

Creates an installable image of the root volume group either in a file or onto a bootable tape.

Syntax

mksysb [**-b** *Number*] [**-e**] [**-i**] [**-m**] [**-p**] [**-v**] [**-V**] [**-X**] *Device* | *File*

Description

The **mksysb** command creates a backup of the operating system (that is, the root volume group). You can use this backup to reinstall a system to its original state after it has been corrupted. If you create the backup on tape, the tape is bootable and includes the installation programs needed to install from the backup.

The file-system image is in backup-file format. The tape format includes a boot image, a bosinstall image, and an empty table of contents followed by the system backup (root volume group) image. The root volume group image is in backup-file format, starting with the data files and then any optional map files.

One of the data files **mksysb** uses is the **/bosinst.data** file. If a **/bosinst.data** file doesn't exist, **/var/adm/ras/bosinst.data** is copied to **/** (root). In AIX 4.3.3 and later versions, **mksysb** always updates the target_disk_data stanzas in **bosinst.data** to match the disks currently in the root volume group of the system where the **mksysb** command is running.

If you are using a customized **/bosinst.data** file and do not want the target_disk_data stanzas updated, you must create the file **/save_bosinst.data_file**. The **mksysb** command does not update **/bosinst.data** if the **/save_bosinst.data_file** exists.

Notes:

1. The image the **mksysb** command creates does not include data on raw devices or in user-defined paging spaces.
2. If you are using a system with a remote-mounted **/usr** file system, you cannot reinstall your system from a backup image.
3. The **mksysb** command may not restore all device configurations for special features, such as **/dev/netbios** and some device drivers not shipped with the product.
4. Some rspc systems for AIX 5.1 and earlier do not support booting from tape. When you make a bootable mksysb image on an rspc system for AIX 5.1 and earlier that does not support booting from tape, the **mksysb** command issues a warning indicating that the tape will not be bootable. You can install a mksysb image from a system that does not support booting from tape by booting from a CD and entering maintenance mode. In maintenance mode you will be able to install the system backup from tape.

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

Flags

-b *Number* Specifies the number of 512-byte blocks to write in a single output operation. When the **backup** command writes to tape devices, the default is 100 for backups by name.

The write size is the number of blocks multiplied by the block size. The default write size for the **backup** command writing to tape devices is 51200 (100 * 512) for backups by name. The write size must be an even multiple of the tape's physical block size.

-e Excludes files listed in the **/etc/exclude.rootvg** file from being backed up. The rules for exclusion follow the pattern matching rules of the **grep** command.

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.

-i Calls the **mkszfile** command, which generates the **/image.data** file. The **/image.data** file contains information on volume groups, logical volumes, file systems, paging space, and physical volumes. This information is included in the backup for future use by the installation process.
Note: Before running the **mkszfile** command, ensure that enough space is available in the **/tmp** file to store a boot image. This space is needed during both backup and installation. To determine the amount of space needed in the **/tmp** file, issue the following command:

```
bosboot -q -a -d device
```

If you use the **-X** flag with the **mksysb** command, you do not need to run the **bosboot** command to determine the amount of space needed in the **/tmp** file.

-m Calls the **mkszfile** command, with the **-m** flag to generate map files.
Note: The use of the **-m** flag causes the functions of the **-i** flag to be executed also.

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

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

-V Verifies a tape backup. This flag causes **mksysb** 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

Device | *File* Specifies the name of the device or file.

Examples

1. To generate a system backup and create an **/image.data** file (generated by the **mkszfile** command) to a tape device named `/dev/rmt0`, type:

   ```
   mksysb -i /dev/rmt0
   ```

2. To generate a system backup and create an **/image.data** file with map files (generated by the **mkszfile** command) to a tape device named `/dev/rmt1`, type:

   ```
   mksysb -m /dev/rmt1
   ```

3. To generate a system backup with a new **/image.data** file, but exclude the files in directory **/home/user1/tmp**, create the file **/etc/exclude.rootvg** containing the line `/home/user1/tmp/`, and type:

   ```
   mksysb -i -e /dev/rmt1
   ```

   This command will backup the **/home/user1/tmp** directory but not the files it contains.

4. To generate a system backup file named **/mksysb_images/node1** and a new **/image.data** file for that image, type:

   ```
   mksysb -i /userimage/node1
   ```

   Note: This file will not be bootable and can only be installed using Network Installation Management (NIM).

5. To generate a system backup on the tape in **/dev/rmt0**, and then verify the readability of file headers, enter:

   ```
   mksysb  /dev/rmt0 -V
   ```

Files

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

Related Information

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

The **/image.data** file.

A procedure to verify the mksysb backup can be found in the article ″Verifying a System Backup″ in the *AIX 5L Version 5.2 Installation Guide and Reference*.

---

mkszfile Command

Purpose

Saves the system state for reinstallation on the current system or another system.

Syntax

mkszfile [**-X**] [**-m**]

Description

Attention: The **mkszfile** command overwrites an existing **/image.data** file with new information.

The **mkszfile** command saves the system state for reinstallation on the current system or on another system. The information saved includes the following:

- System installation information
- Logical volume information for the root volume group
- File system information.

The saved information allows the **bosinstall** routine to recreate the logical volume information as it existed before the backup.

The **mkszfile** command creates the **/image.data** file. The contents of this file are defined by the system in which the image was created. The user can edit the **/image.data** file before calling the **mksysb** command. The **mksysb** command, in turn, only backs up the file systems specified in the **/image.data** file, which reflects the requirements of the **rootvg** file system.

All the saved information is obtained using list commands. The commands are listed in the **/image.data** file as comments for the user's reference when editing this file.

Files on tape cannot be changed. However, in order to override the data files on the tape, the user can create a diskette with the desired files.

The **mkszfile** command checks to be sure there is at least 8MB of free space available in the **/tmp** file system for the boot image.

Note: Before running the **mkszfile** command, ensure that enough space is available in the **/tmp** file to store a boot image. This space is needed during both backup and installation. To determine the amount of space needed in the **/tmp** file, issue the following command:

Flags

-m Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the volume group. This mapping can be used to allocate the same logical-to-physical mapping when the image is restored. The map file locations are stored in the MAPFILE field in the **/image.data** file for each logical volume. Sufficient space would exist in the **/tmp** file system for map creation because the installation routines place the maps in the **/tmp** file system before issuing the **mklv** command.

For example, for the **hd7** logical volume, the location of the map file is **/tmp/vgdata/rootvg/hd7.map**. The MAPFILE field in the **/image.data** file for the **hd7** logical volume is under the entry `MAPFILE=/tmp/vgdata/rootvg/hd7.map`.

The map files in the backup image are copied after the **/bosinst.data** and **/image.data** files.

-X Expands **/tmp** if needed.

Files

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

Related Information

The **mksysb** command.

The **/image.data** file.

---

mktcpip Command

Purpose

Sets the required values for starting TCP/IP on a host.

Syntax

mktcpip { **-S** *Interface* | -h *HostName* **-a** *Address* **-i** *Interface* [**-s**] [**-m** *SubnetMask*] [**-r** *RingSpeed*] [**-t** *CableType*] [**-g** *Gateway*] [**-n** *NameServerAddress* [**-d** *Domain*]] [[**-c** *Subchannel*] **-D** *Destination*] }

Description

The **mktcpip** command sets the required minimal values required for using TCP/IP on a host machine. These values are written to the configuration database. The basic functions of the **mktcpip** command include:

- Setting the host name in both the configuration database and the running machine.
- Setting the IP address of the interface in the configuration database.
- Making entries in the **/etc/hosts** file for the host name and IP address.
- Setting the domain name and IP address of the nameserver, if applicable.
- Setting the subnetwork mask, if applicable.
- Adding a static route to both the configuration database and the running machine, if applicable.
- Starting the specified TCP/IP daemons.

You can use the Network application in Web-based System Manager (wsm) to change network characteristics. You could also use the System Management Interface Tool (SMIT) **smit mktcpip** fast path to run this command.

Flags

-a *Address*
Sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host should have a unique Internet address. The following is the standard format for setting the Internet address:

```
127.10.31.2
```

-c *Subchannel*
Specifies the subchannel address for a System/370 channel adapter.

-D *Destination*
Sets the destination address for a static route. Specify the address in dotted decimal notation. The following is the standard format for setting the destination address for a static route:

```
192.9.52.1
```

-d *Domain*
Specifies the domain name of the name server the host should use for name resolution, if any. The domain name should be in the following format:

```
subdomain.subdomain.rootdomain
```

-g *Gateway*
Sets the gateway address for a static route. Specify the address in dotted decimal notation. The following is the standard format for setting the gateway address for a static route:

```
192.9.52.0
```

-h *Hostname*
Sets the name of the host. If using a domain naming system, the domain and any subdomains must be specified. The following is the standard format for setting the host name:

```
hostname
```

The following is the standard format for setting the host name in a domain naming system:

```
hostname.subdomain.subdomain.rootdomain
```

-i *Interface*
Specifies a particular network interface, for example:

```
tr0
```

-m *SubnetMask*
Specifies the mask the gateway should use in determining the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes, as in the Internet address. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.

-n *NameserverAddress*
Specifies the Internet address of the name server the host uses for name resolution, if applicable. The address should be entered in dotted decimal notation, as follows:

```
127.1.0.1
```

-r *RingSpeed*
Specifies the ring speed for a token-ring adapter. Valid values for the *RingSpeed* variable are either 4- or 16-Mbps.

-S *Interface*
Retrieves information for System Management Interface Tool (SMIT) display.

-s
Starts the TCP/IP daemons.

-t *CableType*
Specifies cable size for Standard Ethernet or IEEE 802.3 Ethernet networks. Valid values for the *CableType* variable are **dix** for thick cable, **bnc** for thin cable, or **N/A** for Not Applicable. The **-t** *CableType* flag should be used only for Standard Ethernet (`en`) and IEEE 802.3 Ethernet (`et`) interfaces.

Examples

To set the required values for starting TCP/IP enter:

```
mktcpip -h fred.austin.century.com -a 192.9.200.4 -i en0 \
-n 192.9.200.1 -d austin.century.com -s
```

Related Information

The **hostname** command, **hostent** command.

The **resolv.conf** file format.

Naming in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

TCP/IP Addressing in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

System Management Interface Tool (SMIT): Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mktun Command

Purpose

Activates tunnel(s).

Syntax

mktun [**-v 4**|**6**] [**-t** *tid_list*] [**-i**] [**-l**]

Description

Use the **mktun** command to activate tunnel(s). For IBM tunnels, this command initiates the security protocol exchanges between the local and the destination host.

Flags

-i	Initiation flag. If the **-i** flag is not used, all the tunnels in the tunnel database (or those listed with the **-t** flag) will be activated. If the **-i** flag is used, only the tunnels whose tunnel definitions in the tunnel database with the status of ″active″ will be activated.
-l	If the **-l** flag is specified, manual tunnels will be activated.
-t	If the **-t** flag is specified, only the tunnel(s) that follows this flag will be activated. If the **-t** flag is not used, all tunnel(s) currently defined in the tunnel database will be activated. The *tid_list* can be a single tunnel ID or a sequence of tunnel IDs separated by ″**,**″ or ″**-**″ (1, 3, 5-7).
-v	The IP version of the tunnels to be activated. The value of **4** specifies IP version 4 tunnels. The value of **6** specifies IP version 6 tunnels. If the **-v** flag is not used, all tunnels for IP version 4 and IP version 6 will be activated.

Related Information

The **chtun** command, **exptun** command, **gentun** command, **imptun** command, **lstun** command, **rmtun** command.

---

mkuser Command

Purpose

Creates a new user account.

Syntax

mkuser [**-R** *load_module*] [**-a**] [*Attribute*=*Value* ...] *Name*

Description

The **mkuser** command creates a new user account. The *Name* parameter must be a unique 8-byte or less string. You cannot use the **ALL** or **default** keywords in the user name. By default, the **mkuser** command creates a standard user account. To create an administrative user account, specify the **-a** flag.

To create a user with an alternate Identification and Authentication (I&A) mechanism, the **-R** flag can be used to specify the I&A load module used to create the user. Users created without the **-R** flag are created locally. Load modules are defined in the **/usr/lib/security/method.cfg** file.

The **mkuser** 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.

You can use the Users application in Web-based System Manager to change user characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkuser** fast path to run this command.

Restrictions on Creating User Names

To prevent login inconsistencies, you should avoid composing user names entirely of uppercase alphabetic characters. While the **mkuser** command supports multi-byte user names, it is recommended that you 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 - (dash), + (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 *Name* parameter cannot contain any space, tab, or new-line characters.

Flags

-a Specifies that the user is an administrator. Only the root user can use this flag or alter the attributes of an administrative user.
-R Specifies the loadable I&A module used to create the user.
load_module

Parameters

Attribute=*Value*	Initializes a user attribute. Refer to the **chuser** command for the valid attributes and values.
Name	Specifies a unique 8-byte or less string.

Security

Access Control: This command should grant execute (x) access only to the root user and members of the security group. 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
rw	**/etc/passwd**
rw	**/etc/security/user**
rw	**/etc/security/user.roles**
rw	**/etc/security/limits**
rw	**/etc/security/environ**
rw	**/etc/group**
rw	**/etc/security/group**
r	**/usr/lib/security/mkuser.default**
x	**/usr/lib/security/mkuser.sys**

Auditing Events:

Event	Information
USER_Create	user

Limitations

Creating a user may not be supported by all loadable I&A modules. If the loadable I&A module does not support creating a user, an error is reported.

Examples

1. To create the `davis` user account with the default values in the **/usr/lib/security/mkuser.default** file, type:

   ```
   mkuser davis
   ```

2. To create the `davis` account with `davis` as an administrator, type:

   ```
   mkuser -a davis
   ```

   Only the root user or users with the UserAdmin authorization can create `davis` as an administrative user.

3. To create the `davis` user account and set the **su** attribute to a value of `false`, type:

   ```
   mkuser su=false davis
   ```

4. To create the `davis` user account that is identified and authenticated through the LDAP load module, type:

   ```
   mkuser -R LDAP davis
   ```

Files

/usr/bin/mkuser	Contains the **mkuser** command.
/usr/lib/security/mkuser.default	Contains the default values for new users.
/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/passwd	Contains password information.
/etc/security/limits	Defines resource quotas and limits for each user.

/etc/security/environ	Contains the environment attributes of users.
/etc/group	Contains the basic attributes of groups.
/etc/security/group	Contains the extended attributes of groups.
/etc/security/.ids	Contains standard and administrative user IDs and group IDs.

Related Information

The **chfn** command, **chgroup** command, **chgrpmem** command, **chsh** command, **chuser** command, **lsgroup** command, **lsuser** command, **mkgroup** command, **mkuser.sys** command, **passwd** command, **pwdadm** command, **rmgroup** command, **rmuser** command, **setgroups** command, **setsenv** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

For more information about administrative roles, refer to Administrative Roles Overview in *AIX 5L Version 5.2 Security Guide*.

---

mkuser.sys Command

Purpose

Customizes a new user account.

Syntax

mkuser.sys *Directory User Group*

Description

The **mkuser.sys** command customizes the new user account specified by the *User* parameter. The **mkuser** command calls the **mkuser.sys** command after it has created and initialized the new account.

The program as shipped creates the home directory specified by the *Directory* parameter, with the owner specified by the *User* parameter, the primary group specified by the *Group* parameter, and a copy of the appropriate profile for the user's shell. The shipped program can be replaced at installation by another program to customize local new-user creation. The installation-specific program should adhere to the error conventions of the supplied program.

Security

Access Control: This command should grant read (r), write (w), and execute (x) access for the root user and members of the security group so the **mkuser** command can execute the program.

Files Accessed:

Mode	File
r	**/etc/passwd**
r	**/etc/security/user**

Files

/usr/lib/security/mkuser.sys	Contains the **mkuser.sys** command.

Related Information

The **mkuser** command.

For more information about the identification and authentication of users, discretionary access control, the trusted computing base, and auditing, refer to the *AIX 5L Version 5.2 Security Guide*.

---

mkvg Command

Purpose

Creates a volume group.

Syntax

mkvg [**-d** *MaximumPhysicalVolumes*] [**-B**] [**-C**] [**-G**] [**-f**] [**-i**] [**-c**] [**-x**] [**-L** *LTGSize*] [**-m** *MaxPvSize*] [**-n**] [**-s** *Size*] [**-t** *factor*] [**-V** *MajorNumber*] [**-y** *VolumeGroup*] *PhysicalVolume ...*

Description

The **mkvg** command creates a new volume group, using the physical volumes represented by the *PhysicalVolume* parameter. After creating the volume group, the **mkvg** command automatically varies on the new volume group using the **varyonvg** command. The exception to this fact is when the volume group is created with the **-C** flag. When the volume group is successfully created, the volume group will not be varied on automatically. Instead, the user must manually **varyon** the volume group.

The **mkvg** command by default creates a volume group that can accommodate 255 logical volumes and 32 physical volumes (disks). These limits can be extended to 512 logical volumes and 128 physical volumes by specifying the **-B** flag.

The **mkvg** command will attempt to determine a proper partition size if none is specified on the command line.

MAXPVS 32 (128 if **-B** flag is used)

Attention: If a big volume is created with the **-B** option, it cannot be imported into AIX 4.3.1 or lower versions.

Notes:

1. The physical volume is checked to verify that it is not already in another volume group. If the system believes the physical volume belongs to a volume group that is varied on, it exits. But if the system detects a description area from a volume group that is not varied on, it prompts the user for confirmation in continuing with the command. The previous contents of the physical volume are lost, so the user must be cautious when using the override function.
2. To use this command, you must either have root user authority or be a member of the **system** group.
3. For disks greater that 4GB, make sure that the Physical Partition Size (**-s**) is set to a large enough value so that 1016 physical partitions per PV limit is not violated. The default value, 4MB, along with the default maximum number of physical partitions (1016), a disk up to 4GB can only be accommodated. Or use large enough factor value (**-t**) such that (*factor* x 1016) is greater than the number of partitions that would need to be created with given partition size and disk size. For example, a partition size of at least 16MB would be needed to create a volume group with a 10GB disk. Or with at factor size of 2, a smaller partition size of 8MB can be used. However, this limits the total number of disks that can be added to the volume group. If a factor value is used, a maximum of MAXPVS/factor disks can be included in the volume group.

4. After AIX 4.1, whenever you create a volume group, the operating system automatically does a varyon. However if you create a volume group with the **-C** flag, the system will not autovaryon the volume group at the end of the Concurrent Capable volume group creation. Instead, the **mkvg** command notifies you to manually **varyonvg** the volume group in either non-concurrent or concurrent mode.
5. This command will fail to add a disk to the volume group if the disk indicates that it is managed by a third party volume manager. To override and clear the disk of the third party volume manger use **chpv -C** *HDiskName*.
6. On AIX 5.2 and later only Enhanced Concurrent Capable volume groups will be created when the **-c** or **-C** flags are specified.

Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a logical volume, adding a physical volume, and so on) might take considerably longer to run.

You can use the Volumes application in Web-based System Manager to change volume characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkvg** fast path to run this command.

Flags

-B Creates a big vg format *volume group*. This can accommodate up to 128 physical volumes and 512 logical volumes.
Note: Because the VGDA space has been increased substantially, every VGDA update operation (creating a logical volume, changing a log ical volume, adding a physical volume, and so on) may take considerably longer to run.

-c Same as **-C** flag. On AIX 5.2 and later only Enhanced Concurrent Capable volume groups will be created.

-C Creates an Enhanced Concurrent Capable volume group. Only use the **-C** flag with the HACMP ES. It is not usable on volume groups and systems not using the HACMP ES product.

Use this flag to create an Enhanced Concurrent Capable volume group containing Serial DASD or SSA disks.

Notes:

1. Enhanced Concurrent volume groups use Group Services. Group Services ships with HACMP ES and must be configured prior to activating a volume group in this mode.
2. Only Enhanced Concurrent Capable volume groups are supported when running with a 64 bit kernel. Concurrent Capable volume groups are not supported when running with a 64 bit kernel.

-d *MaximumPhysicalVolumes* This flag is ignored

-f Forces the volume group to be created on the specified physical volume unless the physical volume is part of another volume group in the Device Configuration Database or a volume group that is active.

-G Similar to **-B** flag except that disk space that can accommodate up to 1024 physical volumes will be reserved at the beginning of the disk. Use this flag if this volume group may ever need be expanded to include more than 128 physical volumes.

-i Reads the *PhysicalVolume* parameter from standard input.

-L *LTGSize* Sets the logical track group size, in number of kilobytes, of the volume group. The value of the *LTGSize* parameter must be 128, 256, 512, or 1024. In addition, it should be less than or equal to the maximum transfer size of all disks in the volume group. The default size is 128 kilobytes.

-m *MaxPvSize*	This flag is ignored.
-n	Specifies that the volume group is not automatically available during a system restart. The default value activates the volume group automatically.
-s *Size*	Sets the number of megabytes in each physical partition, where the *Size* variable is expressed in units of megabytes from 1 through 1024. The *Size* variable must be equal to a power of 2 (example 1, 2, 4, 8). The default value will be the lowest value to remain within the limit of 1016 physical partitions.
-t *factor*	Changes the limit of the number of physical partitions per physical volume, specified by *factor*. The*factor* should be between 1 and 16 for 32 disk volume groups and 1 and 64 for 128 disk volume groups. The maximum number of physical partitions per physical volume for this volume group changes to factor x 1016. The default will be the lowest value to remain within the physical partition limit of *factor* x 1016.
-V *MajorNumber*	Specifies the major number of the volume group that is created.
-x	This flag is ignored. On AIX 5.2 and later only Enhanced Concurrent Capable volume groups will be created.
-y *VolumeGroup*	Specifies the volume group name rather than having the name generated automatically. Volume group names must be unique system wide 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 volume group name created is sent to standard output. The volume group name can only contain the following characters: ″A″ through ″Z,″ ″a″ through ″z,″ ″0″ through ″9,″ or ″_″ (the underscore), ″-″ (the minus sign), or ″.″ (the period). All other characters are considered invalid.

Examples

To create a volume group that contains three physical volumes with partition size set to 1 megabyte, type:

```
mkvg  -s 1 hdisk3 hdisk5 hdisk6
```

The volume group is created with an automatically generated name, which is displayed and available at system restart time.

```
mkvg -s 2 -t 2 -y newvg hdisk1
```

The volume group `newvg` is created with a physical partition size of 2MB and maximum number of physical partitions per physical volume of 2032. The above configuration means that the size of hdisk1 could not be larger than 4064MB (2032*2)

Files

/etc	Directory where the **mkvg** command resides.
/tmp	Directory where the temporary files are stored while the command is running.
/dev	Directory where the character device entry for the volume group is created.

Related Information

The **chvg** command, **lsvg** command, **varyonvg** command.

The Logical Volume Storage Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

The System Management Interface Tool (SMIT) in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

---

mkvgdata Command

Purpose

Creates a file containing information about a volume group for use by the **savevg** and **restvg** commands.

Syntax

mkvgdata [**-X**] [**-m**] *VGName*

Description

The **mkvgdata** command creates a file containing information about a volume group for use by the **savevg** and **restvg** commands. The information includes the list of logical volumes, file systems and their sizes, and the volume group name. One of the following files is created, depending on the type of volume group:

/image.data	Created for information about the root volume group (**rootvg**). The **savevg** command uses this file to create a backup image that can be used by the **bosinstall** routine to reinstall the volume group to the current system or to a new system. The **mkvgdata** command overwrites this file if it already exists. The **/image.data** file is located in the **/** directory.
*vgname***.data**	Created for 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 reinstall the user volume group. The **mkvgdata** command overwrites this file if it already exists. The *vgname***.data** file is located in the **/tmp/vgdata/***vgname* directory, where *vgname* is the volume group name.

The information in either of these files can be edited by the user before issuing the **savevg** command.

Flag

-m	Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the volume group. This mapping can be used to allocate the same logical-to-physical mapping when the image is restored. The map file locations are stored in the MAPFILE field in the **/image.data** file for each logical volume. Sufficient space would exist in the **/tmp** file system for map creation because the installation routines place the maps in the **/tmp** file system before issuing the **mklv** command. For example, for the **hd7** logical volume, the location of the map file is **/tmp/vgdata/rootvg/hd7.map**. The MAPFILE field in the **/image.data** file for the **hd7** logical volume is under the entry `MAPFILE=/tmp/vgdata/rootvg/hd7.map`. The map files in the backup image are copied after the **image.data** or *vgname***.data** files.
-X	Expands **/tmp** if needed.
vgname	Name of volume group to backup.

Files

/image.data	Created when the volume group is **rootvg**.

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

Related Information

The **mkszfile** command, **restvg** command, **savevg** command.

---

mkvirprt Command

Purpose

Makes a virtual printer.

Syntax

mkvirprt [**-A** *AttachmentType*] **-d** *QueueDevice* **-n** *Device* **-q** *PrintQueue* **-s** *DataStream* **-t** *PrinterType* **-T**

mkvirprt -A *AttachmentType*

Description

The **mkvirprt** command creates a virtual printer definition and assigns it to the specified print queue and queue device. A virtual printer definition is a set of attribute values that describe a particular data stream for a particular printer. Before a print job can be queued with the **enq** command, **qprt** command, **lp** command, or **lpr** command, a virtual printer definition must be created for the printer's print queue and queue device.

Printers that support only one printer data stream, such as the 4201-3 Proprinter III, need only one virtual printer defined. Printers that support multiple printer data streams, such as the IBM 4216-31 Page Printer II, need a virtual printer defined for each data stream.

To create a virtual printer definition for a printer attached to an ASCII terminal, use the **-T** flag with the **mkvirprt** command.

After a virtual printer definition is created, its attribute values can be displayed with the **lsvirprt** command and changed with the **chvirprt** command.

The **mkvirprt** command becomes interactive if only the -A flag is specified with the command. Prompts are issued requesting the necessary parameter values. Prerequisite spooler queues and spooler queue devices are generated automatically, and all virtual printer definitions needed for the printer are defined with a single invocation of the **mkvirprt** command for the specified attachment type.

When the first prompt asks for a device name, if the device name entered is not that of a printer, or if an * (asterisk) precedes the device name, a list of printers is displayed. Otherwise, the printer type is assumed to be the same as that of the device.

Also, when a prompt asks for a print queue name, the queue name entered may optionally be followed by a colon and a queue device name. If no queue device name is provided, the queue device name is assumed to be the same as the device name.

> **Note:** Queue and device names must begin with an alphabetic character.

You can use the Printer Queues application in Web-based System Manager (wsm) to change printer characteristics. You could also use the System Management Interface Tool (SMIT) **smit mkvirprt** fast path to run this command.

Flags

-A *AttachmentType*
Specifies the type of printer attachment. The most common values for the *AttachmentType* variable value are:

Attachment Type	Represents
local	Locally connected printers
remote	Remote print queues
ascii	Printers attached to an ASCII terminal
file	Print output redirected to a regular file.

This flag is optional, and if the **-A** flag is not specified the default attachment type is **file**. If the **-A** flag is the only flag specified on the command line, the **mkvirprt** command goes into interactive mode and executes steps specified in the corresponding **.config** file.

-d *QueueDeviceName*
Specifies the name of an existing queue device to which the virtual printer is assigned.

-n *DeviceName*
Specifies the name of the printer device. Device names include **lp0** for printer 0, **lp1** for printer 1, and so on.

-q *PrintQueueName*
Specifies the special file name of an existing print queue to which the virtual printer is to be assigned. Note that you do not have to specify the path name to the file, such as the **/dev/lp0** file, you just need to specify **lp0**.

-s *DataStreamType*
Specifies the printer data stream type. Data stream types include:

Type	Description
asc	Extended ASCII
ps	PostScript
pcl	Hewlett-Packard PCL
630	Diablo 630
855	Texas Instruments 855
gl	Hewlett-Packard GL
kji	Kanji

-t *PrinterType*
Specifies the printer type. Printer types include 4201-3**,** ti2115**,** and so on. For more information on available printer types, see ″Printer Support Provided″ and ″Printer Specific Information″ in *AIX 5L Version 5.2 Guide to Printers and Printing*.

-T
Specifies that the printer is attached to an ASCII terminal.

Examples

1. To make a virtual printer for the `asc` printer data stream for the 4029 printer attached locally, enter:
   ```
   mkvirprt -A local -d mypro -n lp0 -q proq -s asc -t 4019
   ```
2. To make a virtual printer for a printer connected to an ENA 4033 network adapter, and to be prompted for the parameter values, enter:
   ```
   mkvirprt -A ena
   ```

Files

/usr/sbin/mkvirprt	Contains the **mkvirprt** command.
/etc/qconfig	Contains configuration files.
/usr/lib/lpd/pio/predef/*	Contains predefined printer attribute files.
/var/spool/lpd/pio/@local/custom/*	Contains customized virtual printer attribute files.

/usr/lib/lpd/pio/etc/*.attach	Contains attachment type files.
/usr/lib/lpd/pio/etc/*.config	Contains the configuration file for the printer.
/var/spool/lpd/pio/@local/ddi*	Contains digested virtual printer attribute files.

Related Information

The **chvirprt** command, **lp** command, **lpr** command, **lsvirprt** command, **mkque** command, **mkquedev** command, **qprt** command, **rmvirprt** command.

The **/etc/qconfig** file.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Printer Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Spooler Overview for System Management in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Specific Information in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Support in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Virtual Printer Definitions and Attributes in *AIX 5L Version 5.2 Guide to Printers and Printing*.

Printer Addition Management Subsystem: Programming Overview in *AIX 5L Version 5.2 Kernel Extensions and Device Support Programming Concepts*.

---

mm Command

Purpose

Prints documents formatted with memorandum macros.

Syntax

mm [**-M** *Media*] [**-c**] [**-e**] [**-E**] [**-t**] [**-12**] [**-T***Name*] { *File* ... | **-** }

Description

The **mm** command formats documents that use the **nroff** command and the **mm** macro package. The **mm** command has flags that specify preprocessing by the **tbl** and **neqn** commands and postprocessing by various terminal-oriented output filters. The proper pipelines and the required flags for the **nroff** command are generated depending on the flags that are selected.

Notes:

1. Use the **-o***List* flag of the **nroff** command to specify ranges of output pages. Remember that if the **mm** command is called with the **-e**, **-t**, or **-** (minus sign) flags together with the **-o***List* flag, and if the last page of the document is not specified by the *List* variable, you may receive a `broken pipe` message. This message is not an indication of any problem and can be ignored.
2. The **mm** command calls the **nroff** command with the **-h** flag. With this flag, the **nroff** command assumes that the workstation has tabs set every 8 character positions.
3. If you use the **-s** flag of the **nroff** command (to stop between pages of output), use a linefeed (rather than the Enter key or a newline character) to restart the output. The **-s** flag of the **nroff** command does not work with the **-c** flag of the **mm** command or if the **mm** command automatically calls the **col** command**.**

4. Providing inaccurate information to the **mm** command about the kind of workstation its output is to be printed on will produce unsatisfactory results. However, if you are redirecting output to a file, use the **-T37** flag. Then, use the appropriate workstation filter when you print the file.

To obtain a list of **mm** command flags, enter the command name with no parameters. The flags can occur in any order, but they must come before the *File* parameter. Any other flags (for instance, **-r***ANumber*) are passed to the **nroff** command.

Flags

-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 8.27 X 11.69 inches (210 X 297 mm).

B5 Specifies a paper size of 6.93 X 9.84 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.

-c Calls the **col** command. Note that the **col** command is called automatically by the **mm** command for the following terminal names. The following devices can be specified by the **-T***Name* flag, the **$TERM** shell variable, or by using the default:

- **ppds**
- **lp**
- **2631**
- **8510**

-e Calls the **neqn** command; also causes the **neqn** command to read the **/usr/share/lib/pub/eqnchar** file. See the **eqnchar** file format.

-E Calls the **-e** flag of the **nroff** command.

-t Calls the **tbl** command.

-12 Uses 12-pitch font. Use this when the **$TERM** shell variable is set to 300, 300s, 450, or 1620. (The pitch switch on the DASI 300 and 300s workstations must be manually set to 12 if this flag is used.)

-T*Name* Uses the workstation type specified by the *Name* variable.

By default, the **mm** command uses the value of the **$TERM** shell variable from the environment as the value of the *Name* variable. If the **$TERM** shell variable is not set, the **mm** command uses **lp** (the generic name for printers that can underline and tab). If several workstation types are specified, the last one listed applies.

- Forces input to be read from standard input.

Parameters

File Specifies the file that the **mm** command formats.

Examples

1. When the **$TERM** shell variable is set in the environment to the **hplj** command, the following two command lines are equivalent:

```
mm -t -rC3 File
tbl File | nroff -mm -Thplj -h -rC3
```

2. The **mm** command reads the standard input when you specify a **-** (minus sign) flag instead of a value for the *File* variable. This option allows you to use the **mm** command as a filter, as follows:

```
cat File | mm -
```

Note: Using other files together with a **-** (minus sign) flag leads to undesired results.

Environment Variables

$TERM Specifies the terminal names.

Files

/usr/share/lib/pub/eqnchar	Contains special character definitions for the **eqn** command and the **neqn** command.

Related Information

The **col** command, **env** command, **eqn** command, **greek** command, **hplj** command, **mmt** command, **neqn** command, **nroff** command, **tbl** command.

The **eqnchar** file, **profile** file.

The **nterm** file format describes terminal driving tables for the nroff command.

The article ″mm Macro Package for the mm, mmt, nroff, and troff Commands″ in the **troff** Command.

---

mmt Command

Purpose

Typesets documents.

Syntax

mmt [**-M** *Media*] [**-a**] [**-c**] [**-e**] [**-g**] [**-p**] [**-t**] [**-z**] [**-T***Name* | **-D***Destination*] [*File* | **-**]

Description

Similar to the **mm** command, the **mmt** command typesets its input using the **troff** command, rather than formatting it with the **nroff** command. The **mmt** command uses the **mm** macro package. There are flags to specify preprocessing by the **tbl**, **pic**, **eqn**, and **grap** commands. The proper pipelines, required parameters, and flags for the **troff** command and the **mm** macro package are generated, depending on the flags selected.

There are several flags that are specific to the **mmt** command. Any other parameters or flags (for instance, **-r***ANumber* or **-a**) that you give the **mmt** command are passed to the **troff** command. You can put flags in any order, but they must be listed before any input files. *File* specifies the file that the **mmt** command formats. If you do not give *File* parameters or other flag variables, the **mmt** command prints a list of its flags.

The **mmt** command, unlike the **troff** command, automatically pipes its output to a postprocessor, unless specifically requested not to do so. The user should not specify a postprocessor when using the **mmt** command. The precedence is as follows:

1. The **-z** flag; no postprocessor is used.
2. The **-T***Name* flag.
3. The **TYPESETTER** environment variable is read.
4. The default is set to **ibm3816**.

The **mmt** command reads standard input when you specify a **-** (minus sign) instead of any *File* parameters.

Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output.

> **Note:** If you call the **mmt** command with one or more of the **-e**, **-c**, **-t**, **-p**, **-g**, and - (minus sign) flags together with the **-o***List* flag of the **troff** command, you may receive a `broken pipe` message if the last page of the document is not specified by the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

Flags

-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 8.27 X 11.69 inches (210 X 297 mm).

A5 Specifies a paper size of 5.83 X8.27 inches (148 X 210 mm).

B5 Specifies a paper size of 6.93 X 9.84 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.

-a Displays readable **troff** output to the terminal.

-c Preprocesses the input files with the **cw** command.

-e Calls the **eqn** command; also causes the **eqn** command to read the **/usr/share/lib/pub/eqnchar** file (see the **eqnchar** file format).

-g Calls the **grap** command, which in turn calls the **pic** command.

-p Calls the **pic** command.

-t Calls the **tbl** command.

-z Starts no output filter to process or redirect the output of the **troff** command.

-D*Destination* Directs the output to a device specified by the *Destination* variable. Supported destination devices for English-language output is 4014, which is the Tektronix 4014 terminal by way of the **tc** command.

-T*Name* Creates output for a **troff** device as specified by the *Name* variable. The output is sent through the appropriate postprocessor.. The default value is **ibm3816**. Possible Name variables are:

ibm3812
3812 Pageprinter II.

ibm3816
3816 Pageprinter.

hplj Hewlett-Packard LaserJet II.

ibm5587G
5587-G01 Kanji Printer multi-byte language support.

psc PostScript printer.

X100 AIXwindows display.

- Forces input to be read from standard input.

Related Information

The **cw** command, **eqn** command, **grap** command, **mm** command, **mvt** command, **pic** command, **tbl** command, **tc** command, **troff** command.

The **eqnchar** file format contains special character definitions for the eqn and neqn commands.

The article ″mm Macro Package for the mm, mmt, nroff, and troff Commands″ in the **troff** Command.

---

mmtu Command

Purpose

Displaying, adding, and deleting maximum transfer unit (MTU) values used for path MTU discovery. This command only applies to AIX 4.2.1 or later.

Syntax

mmtu { **-a** *Value* | **-d** *Value* | **-s** }

Description

Use the **mmtu** command to display, add, and delete maximum transfer unit (MTU) values to the list of potential path MTU values. Path MTU discovery uses the list of potential path MTU values to detect the path MTU. The list of potential path MTU values is only used when there are routers in the path that do not comply with RFC 1191. The user must have administrative authority to add or delete MTU values.

Flags

-a *Value*	Adds the new MTU to the list of potential path MTU values.
-d *Value*	Deletes the value from the list of potential path MTU values.
-s	Displays the current list of potential path MTU values.

Examples

1. To add a value to the list of potential path MTU values, enter:

   ```
   mmtu -a mtu-value
   ```

2. To delete a value from the list of potential path MTU values, enter:

   ```
   mmtu -d mtu-value
   ```

3. To display the contents of the list of potential path MTU values, enter:

   ```
   mmtu -s
   ```

Files

/usr/sbin/mmtu	Contains the **mmtu** command.

Related Information

The **netstat** command, **no** command.

---

mobip6ctrl Command

Purpose

Configures and manages the mobile IPv6 home agent and correspondent node functionality.

Syntax

mobip6ctrl [**-R**] [**-b**] [**-S** { **0** | **1** }] [**-n** { **0** | **1** }] [**-l** *LifeTime*] [**-a** | **-d** *HomeAddress CareOfAddress MyAddress*]

Description

The **mobip6ctrl** command is used to configure and manage the mobile IPv6 home agent and correspondent node. It can enable and disable NDP proxy and IP security checking, and it can be used to display or modify the mobile IPv6 binding cache.

NDP proxy must be enabled if the system is configured as a home agent. This allows the home agent to intercept packets addressed to mobile nodes that are not currently on their home network.

IP security checking enables checking to ensure that IP security is used for the Binding Update and Binding Acknowledgement messages sent for mobile IPv6. Because these two types of messages have the ability to affect the routing of packets addressed to a mobile node, they would represent a significant security vulnerability if not protected by IP security. If checking is enabled, the mobile IPv6 home agent or correspondent node will discard any Binding Update or Binding Acknowledgement packets that are not protected by IP security.

The mobile IPv6 binding cache on a home agent or correspondent node maps home addresses to the current care-of addresses for each mobile node. This allows the home agent to tunnel traffic to the mobile node at its current location, and allows a correspondent node to send packets directly to a mobile node at its current location. The **mobip6ctrl** command can be used to view the binding cache or manually edit it for debugging purposes.

Normally, this command is used from the **/etc/rc.mobip6** script when mobile IPv6 has been configured using system management.

Flags

-a *HomeAddress CareOfAddress MyAddress*	Adds this entry to the binding cache.
-b	Displays all binding cache entries.
-d*HomeAddress CareOfAddress MyAddress*	Delete this entry from the binding cache.
-l *LifeTime*	Specifies the default life time value for binding cache entries in seconds.
-n 0 | **1**	Activates or deactivates NDP proxy capabilities. A value of 1 activates the NDP proxy capabilities, and a value of 0 disables NDP proxy capabilities. The default value is 0.
-R	Resets all the binding cache entries.
-S 0 | **1**	Enables or disables checking to ensure that IP security is used for all Binding Update and Binding Acknowledgement packets. A value of 1 enables checking, and a value of 0 disables checking. The default value is 0.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example enables NDP proxy and IP security checking for mobile IPv6:

   ```
   mobipctrl -S 1 -n 1
   ```

2. The following example displays all entries in the binding cache:

   ```
   mobip6ctrl -b
   ```

   The output from this command looks similar to the following:

   ```
                  BINDING CACHE LIST (1 elem)

   Home Address.........: 3ffe:300:20:1102::217
   Care-Of Address......: 3ffe:300:20:1101::217
   My Address...........: 3ffe:300:20:1102::223
   Life time............: 518
   Time since last usage: 50
   Rate limit time......: 0
   Retransmit count.....: 0
   Sequence number......: 14
   Registered by me.....: 1
   Prefix length........: 64
   ```

Related Information

The **kmodctrl** command, **mobip6reqd** command, **ndpd-router** command, **rc.mobip6** command.

The Mobile IPv6 section in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

mobip6reqd Daemon

Purpose

Provides the Mobile IPv6 home agent daemon.

Syntax

To run the daemon using the System Resource Controller:

startsrc -s mobip6reqd

To run the daemon without using the System Resource Controller:

mobip6reqd

Description

The **mobip6reqd** daemon must be running in order for the system to function as a mobile IPv6 home agent. This daemon enables the home agent to perform NDP proxying for mobile nodes. The daemon is normally started automatically by the **/etc/rc.mobip6** script if the mobile IPv6 home agent has been enabled using system management.

Exit Status

0 The command completed successfully.

>0 An error occurred.

Security

You must be the root user or a member of the system group to execute this command.

Examples

1. The following example starts the **mobip6reqd** daemon:

   ```
   startsrc -s mobip6reqd
   ```

2. The following example stops the **mobip6reqd** daemon:

   ```
   stopsrc -s mobip6reqd
   ```

Related Information

The **kmodctrl** command, **mobip6ctrl** command, **ndpd-router** command, **rc.mobip6** command.

The Mobile IPv6 section in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

---

monacct Command

Purpose

Performs monthly or periodic accounting.

Syntax

/usr/sbin/acct/monacct [*Number*]

Description

The **monacct** command performs monthly or periodic accounting. The intervals are set in the **crontab** file. You can set the **cron** daemon to run the **monacct** command once each month or at some other specified time period. The **monacct** example shows how to set up this command for use with the **cron** daemon. See the **crontab** command for more information about setting up **cron** files.

The *Number* parameter indicates which month or other accounting period to process. The default value of the *Number* parameter is the current month. The **monacct** command creates summary files in the **/var/adm/acct/fiscal** file and restarts summary files in the **/var/adm/acct/sum** file, the cumulative summary to which daily reports are appended.

> **Note:** You should not share accounting files among nodes in a distributed environment. Each node should have its own copy of the various accounting files.

Security

Access Control: This command should grant execute (x) access only to members of the administrative group.

Example

To produce automatically a monthly accounting report, add the following to the **/var/spool/cron/crontabs/root** file:

```
15 5 1 * * /usr/sbin/acct/monacct
```

This example shows the instructions that the **cron** daemon will read and act upon. The **monacct** command will run at 5:15 (`15 5`) the first day of each month (`1`). This command is only one of the

accounting instructions normally given to the **cron** daemon. See ″Setting Up an Accounting System″ in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices* for more information on typical **cron** accounting entries.

Files

/usr/sbin/acct	Contains the accounting commands.
/var/adm/acct/fiscal	Contains accounting data files.
/var/adm/acct/sum	Cumulative daily accounting records.
/var/spool/cron/crontabs	Contains the commands to be run by the **cron** daemon at regularly scheduled intervals.

Related Information

The **acctcms** command, **prtacct** command, **acctmerg** command, **crontab** command.

The **cron** daemon.

For more information about the Accounting System, the preparation of daily and monthly reports, and the accounting files, see the Accounting Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Setting Up an Accounting System in *AIX 5L Version 5.2 System Management Guide: Operating System and Devices* explains the steps you must take to establish an accounting system.

---

mon-cxma Command

Purpose

Monitor status of 128-port asynchronous subsystem and attached devices.

Syntax

To Display All 128-Port Adapters:

mon-cxma

To Display Syntax or Slots and Bus Information:

mon-cxma { **-h** | **-x** }

To Display Specific Slots and Bus Information:

mon-cxma { [**-l** [*LogFile*] [**-f** [*DeviceFile*]] [**-s** [*SlotNumber*]] [**-b** [*BusNumber*]] }

Description

The **mon-cxma** command is a software tool which provides a means to monitor the status of serial devices and remote async nodes (RAN) attached to the IBM 128-port asynchronous adapter. It is used for subsystem problem determination and can be accessed locally and remotely via modem. The only restriction on modem access is that the modem can not be physically attached to the 128-port adapter being monitored.

When the user enters the **mon-cxma** command at the command line, it automatically detects and displays all available 128-port adapters in the system. The bus and slot location within the system is displayed for each adapter and the user can select adapter to monitor.

You can use the Software application in Web-based System Manager (wsm) to run this command. You could also use the System Management Interface Tool (SMIT) **smit 128psync** fast path to advance

directly to the ″128-Port Asynchronous Adapter″ menu. When run from SMIT, the **mon-cxma** command automatically displays all available 128-port adapters in the system. Refer to ″Running the Monitor from SMIT″ in *AIX 5L Version 5.2 Asynchronous Communications Guide* for more information.

Flags

-b [*BusNumber*]	Specifies the bus number of the device. Valid values for *BusNumber* are 0 to (n-1), where n is the number of buses the system has.
-f [*DeviceFile*]	Specifies the device special file. Use this file to look at a specific device driver without having to make a selection. The default device special file is **/dev/cxma0**.
-h	Shows syntax information.
-l [*LogFile*]	(Lowercase L) Specifies the file to be used as the log. Use this file to store information from the screen when the IMAGE key is pressed. The default log file is **/tmp/mon-cxma.log**.
-s [*SlotNumber*]	Specifies the slot number of the device. Valid values for *SlotNumber* are 0 to (n-1), where n is the number of slots the system has.
-x	Shows the POS (Programmable Select Option) register values for all the slots and buses.

Note: -x and **-h** ignore other options.

Security

Access Control: Root authority required to run this command.

Auditing Events: N/A

Examples

1. To run the mon-cxma command using the SMITfastpath, enter:
   ```
   smit 128psync
   ```
2. To display all 128-port adapters, enter:
   ```
   /usr/lbin/tty/mon-cxma
   ```

Files

/usr/lbin/tty/mon-cxma	Contains the **mon-cxma** command.
/tmp/mon-cxma.log	Contains the log file.

Related Topics

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

---

monitord Daemon

Purpose

Communicates with the License Use Management server and requests a concurrent-use license for each countable login.

Syntax

monitord [**-t** *Minutes*] [**-v** *Version.Release*]

Description

The operating system has multiple ways to access the system, and each of them has a different behavior upon exit. The **monitord** daemon provides a common interface to the License Use Management **netlsd**. **monitord** communicates with the License Use Management server and requests a concurrent-use license for each countable login.

> **Note:** The License Use Management licensing mechanism is used only if the system has the *floating license mode* enabled.

After user logout, **monitord** requests **netlsd** to release the specific license the user was using, in order to make it available for further logins.

monitord is started when the **chlicense -f on** command is used to enable the *floating license mode*. When the *floating license mode* is enabled, **monitord** is started upon system startup via an entry in **/etc/inittab**. The default (invoked without **-t** option) is an interval of fifteen minutes.

The entry in **/etc/inittab** looks like the following:

```
monitord:2:once:/usr/sbin/monitord >/dev/console 2>&1
```

Flags

-t *Minutes*	Sets the value in minutes of the heartbeat interval. A value of 0 sets an infinite interval. The default is fifteen minutes.
-v *Version***.***Release*	Enables the *floating license mode* for a license of the specified *Version* and *Release*.

---

moo Command

Purpose

Starts the number-guessing game.

Syntax

moo

Description

The **moo** command picks a combination of four random, non-repeating numbers. You guess four numbers at the `your guess?` prompt. Each correct number in an incorrect position in the four number combination scores ″cow.″ Each correct number in the correct position in the four number combination scores a ″bull.″ For example:

```
your guess?
1470
bulls = 0  cows = 1
your guess?
```

In this example, one of the four numbers (1, 4, 7, and 0) is correct but in the incorrect position. None of the numbers are correct and in the correct position.

To quit the game, press the Interrupt (Ctrl-C) or End Of File (Ctrl-D) key sequence.

File

/usr/games	Contains the system's games.

Related Information

The **arithmetic** command, **back** command, **bj** command, **craps** command, **fish** command, **fortune** command, **hangman** command, **number** command, **quiz** command, **ttt** command, **turnoff** command, **turnon** command, **wump** command.

---

more Command

Purpose

Displays the contents of files one screen at a time.

Syntax

more [**-c**] [**-d**] [**-e**] [**-i**] [**-l**] [**-N**] [**-s**] [**-u**] [**-v**] [**-z**] [**-n** *Number*] [**-p** *Subcommand*] [**-t** *Tagstring*] [**-W** *Option*] [**-x** *Tabs*] [*File* ...]

Description

The **more** command reads files and displays the text one screen at a time. The command pauses after each screen and prints the word `More` at the bottom of the screen. If you then press a carriage return, the **more** command displays an additional line. If you press the space bar, the **more** command displays another full screen of text.

> **Note:** On some terminal models, the **more** command clears the screen, instead of scrolling.

Instead of naming files to read, you can either redirect or pipe standard output, such as a long directory listing, to the **more** command. The command adds a % (percent sign) to its prompt when reading from a file rather than a pipe. This provides the percentage of the file (in characters, not lines) that the **more** command has read.

The **more** command sets the terminal to NOECHO mode so the output can be continuous. With the exception of the / and ! subcommands, commands that are typed do not normally show up on the terminal. If the standard output is not a terminal, the **more** command will act just like the **cat** command, except that a header will be printed before each file in a series.

Environment Variables

Environment variables affect the way the **more** command works. You can set some environment characteristics in the **/etc/environment** file and system profile files, such as the **.ksh**, **.csh**, and **.profile** files. Refer to ″User Environment and System Information Overview″ in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* for discussions about determining and configuring your system environment.

The **more** command uses the **TERM** variable to determine terminal characteristics. If this variable is NULL or not set, the command uses the default terminal type. The **/usr/share/lib/terminfo** directory contains definitions for terminal characteristics.

By default, the **more** command window size is 2 lines less than what the system terminal is capable of. The command sets the default window size based on the **LINES** variable. Also, you can easily adjust the window size for each run of the command by adding the **-n** flag.

Use the **MORE** variable to customize the **more** command with your preferred configuration each time the system starts. This variable accepts **more** command flags.

Flags

-c Prevents the screen from scrolling, which makes text easier to read as the **more** command writes to the screen. The system ignores the **-c** flag if the terminal cannot clear to the end of a line.

-d Prints a message, appended to the `More` prompt at the bottom of the screen, about which keys continue, quit, and provide help for the **more** command. Displays error messages rather than ringing the terminal bell if an unrecognized command is used. This is helpful for inexperienced users.

-e Exits automatically after displaying the last line of the last file.

-i Searches for patterns without considering uppercase and lowercase.

-l Pauses after detecting a page break in the input. If the `-l` flag is not used, the **more** command pauses to accept commands after any line containing a `^L` (CTRL-L) character. Also, if a file begins with a FORMFEED, the screen is cleared before the file is printed.

-N Suppresses line numbering. The default display, with line numbers, can slow the **more** command's performance on very large input files. The line numbering feature displays the line number in the **=** subcommand and passes the line number to the editor (if it is the **vi** editor).

-n *Number* Configures the **more** command to display the specified number of lines in the window. Without the **-n** flag, the **more** command defaults to two lines less than what the terminal is capable of. For example, on a 24-line terminal, the default is 22 lines. The `-n` option overrides any values obtained from the environment.

-p *Subcommand* Starts the **more** command and specified subcommand for each *File* operand. For example, **more -p 50j text1 text2** displays the **text1** file at the fiftieth line; then does the same for the **text2** file when you finish the first. See ″Subcommands″ for descriptions of **more** subcommands.

If the command is a positioning command, such as a line number or a regular expression search, set the current position to represent the final results of the command, without writing any intermediate lines of the file. For example, the two commands:

```
more -p 1000j filename

more -p 1000G filename
```

are functionally the same and will start the display with the current position at line 1000, passing the lines that j would write and would scroll off the screen if it had been issued during the file examination.

If the positioning command is unsuccessful, the first line in the file will be the current position.

-s Reduces multiple blank lines in the output to only one blank line. The **-s** flag is particularly helpful in viewing output from the **nroff** command.

-t *Tagstring* Displays the portion of the file that contains the specified tag. This flag works only on files containing tags created with the **ctags** command.

-u Prevents the **more** command from treating a backspace character as a printable control character (displayed as a ^H (CTRL-H)), suppressing backspacing, underlining, or creating reverse video text for underlined information in a source file. The **-u** flag also forces the **more** command to recognize a carriage-return character, if it exists, at the end of a line.

-v Suppresses the graphical translation of nonprinting characters. Without the **-v** flag, the **more** command graphically interprets all non-ASCII and most control characters, except Tab, Backspace, and Return. For example, if you do not use the **-v** flag, the **more** command displays the non-ASCII characters Ctrl-x as `^X` and x as `M-x`.

-W *Option* Provides the specified *Option* to the **more** command as an extension:

notite Prevents the **more** command from sending the terminal initialization string (either the **ti termcap** or the **smcup terminfo** capability) before displaying the file. This option also prevents the **more** command from sending the terminal de-initialization string (either the **te termcap** or the **rmcup terminfo** capability) before exiting.

tite Causes the **more** command to send the initialization and de-initialization strings. This is the default.

These options control whether the **more** command sends the initialization strings described, which, for certain terminals (such as some xterms), cause the **more** command to switch to an alternative screen. The effect of switching screens is to erase the display of the file you were viewing.

-x *Tabs* Sets tab stops at the specified *Tabs* position. The default tab setting is 8 columns.

-z Graphically displays the Tab, Backspace, and Return control characters. With the **-z** flag, the **more** command translates the Backspace character as `^H`, Return as `^M`, and Tab as `^I`.

Subcommands

The **more** command accepts subcommands when the command pauses and as parameters for the **-p** flag. Many subcommands take an optional integer, symbolized here by *K*, which you must enter before the subcommand, with no space between. The **more** command, in the paused state, processes subcommands immediately and does not require you to press the Enter key.

The **more** command uses the following subcommands:

h Displays a help screen that describes the **more** subcommands.

v Starts the **vi** editor, editing the current file in the current line.

r or **^L** Refreshes the display.

R Refreshes the display and removes buffered input.

[*K*](Spacebar) Moves forward *K* lines when you press the spacebar. If you do not give a value for *K*, pressing the spacebar displays the next full screen by default. This spacebar subcommand is the same as [*K*]**f** or [*K*]**^F** or [*K*]**z**.

[*K*]**f** or [*K*]**^F** or [*K*]**z**

Moves forward *K* lines, or one full screen if you do not give a value for *K*.

[*K*]**b** or [*K*]**^B** Moves backward *K* lines, or one full screen if you do not give a value for *K*.

[*K*]**d** or [*K*]**^D** Moves forward *K* lines, or half a screen if you do not give a value for *K*. If you give a value for *K*, the **more** command sets the **d** and **u** scroll size to *K* lines for the session.

[*K*]**u** or [*K*]**^U** Moves backward *K* lines, or half a screen if you do not give a value for *K*. If you give a value for *K*, the **more** command sets the **d** and **u** scroll size to *K* lines for the session.

[*K*]**j** or [*K*](Enter) or [*K*]**^E**

Moves forward *K* lines, or one line if you do not give a value for *K*.

[*K*]**k** or [*K*]**^Y** Moves backward *K* lines, or one line if you do not give a value for *K*.

[*K*]**g** Moves to the beginning of the file, unless you give a line number for *K*. The default for *K* is line number 1.

[*K*]**G** Moves to the last line in the file, unless you give a line number for *K*. The default for *K* is the last line in the file.

[*K*]**p** or [*K*]**%** Moves to the point in the file that is *K* percent of the total file. The default for *K* is one percent, or the first line in the file.

m*a-z* Marks the current position in the file with the specified letter.

'*a-z*	(Single quote) Moves to the position marked with the specified letter.
''	(Two single quotes) Returns to the position from which the last large movement (moving more than a page) command was run. If no such movements have been made, returns to the beginning of the file.
[*K*]**/***pattern*	(Slash) Searches forward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**/!***pattern*	(Slash, exclamation mark) Searches forward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**?***pattern*	(Question mark) Searches backward, from the current position, for the specified occurrence of the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**?!***pattern*	(Question mark, exclamation mark) Searches backward, from the current position, for the specified occurrence of a line that does not contain the specified pattern of characters. The default value for *K* is the first occurrence.
[*K*]**n**	Repeats the last search, specifying an occurrence of the pattern (or an occurrence *not* containing the pattern if the search subcommand included **!**). The default value for *K* is the first occurrence.
:a	Lists the file or files you named in the **more** command line.
:f or **^G** or **=**	Displays information about the current file: • file name • order of the file in the list of files • current line number • current position in the file, given as a percentage • current byte number and total bytes to display.
:e[*File*] or **E**[*File*]	Examines the specified file, provided you named it in the **more** command line.
[*K*]**:n** or [*K*]**N**	Examines either the next file (if you do not give a value for *K*) or the file *K* number of positions forward in the list of files you named in the **more** command line.
[*K*]**:p** or [*K*]**P**	Examines either the previous file (if you do not give a value for *K*) or the file *K* number of positions backward in the list of files you named in the **more** command line.
:t *Tagstring*	Displays the portion of the file that contains the specified tag. This subcommand works only on files containing tags created with the **ctags** command. The **:t** subcommand is the interactive version of the **-t** flag.
:q or **q** or **Q**	Exits the **more** command.
:!*command* or **!***command*	Starts the specified command in a new shell.

Exit Status

This command returns the following exit values:

0	Successful completion.
>0	An error occurred.

Examples

1. To view a file named `myfile`, enter:
   ```
   more myfile
   ```
2. To view output from the **nroff** command, enter:
   ```
   ls -l | more
   ```
3. To view each file starting at its last screen, enter:
   ```
   more -p G file1 file2
   ```
4. To view each file with the 100th line at the current position, enter:
   ```
   more -p 100 file1 file2
   ```

   Typically, the current position in a **more** command display is the third line on the screen. In this example, the first line on the screen is the 98th line in the file.
5. To view each file starting with the first line that contains the `foo` string, enter:
   ```
   more -p /foo file1 file2
   ```

   The **more** command displays the line in the current position, the third line on the screen.

Files

/usr/share/lib/terminfo	Indicates the terminal information database.

Related Information

The **cat** command, **csh** command, **ctags** command, **ksh** command, **pg** command, **script** command.

The **environment** file, **terminfo** file.

User Environment and System Information Overview in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Understanding Locale Environment Variables in the *AIX 5L Version 5.2 National Language Support Guide and Reference*.

Shells Overview in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Input and Output Redirection Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Files Overview in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

File and Directory Access Modes in the *AIX 5L Version 5.2 System Management Guide: Operating System and Devices*.

---

mosy Command

Purpose

Converts the ASN.1 definitions of Structure and Identification of Management Information (SMI) and Management Information Base (MIB) modules into objects definition files for the **snmpinfo** command.

Syntax

mosy -o *OutputFile* [**-s**] *InputFile* ...

Description

The **mosy** command reads in the ASN.1 definitions of SMI and MIB modules and produces objects definition files in specific formats. The resulting objects definition files are used by the **snmpinfo** command.

The *InputFile* parameter files are required to be in the **smi.my** or **mibll.my** format. Sample files are the **/usr/samples/snmpd/smi.my** and **/usr/samples/snmpd/mibll.my** files. See the **smi.my** and the **mibll.my** files for information on the required format of the file specified by the *InputFile* parameter.

The **mosy -o** command is used to create the objects definition file specified by the *OutputFile* parameter for the **snmpinfo** command. This file is normally the **/etc/mib.defs** file.

The objects definition file can be created with one pass of the **mosy** compiler if the **smi.my** and **mibll.my** files are both specified as *InputFile* parameters. The **smi.my** file must precede the **mibll.my** file on the command line.

The **mosy -o** command can also be used to create subfiles. If subfiles are created separately from the SMI and MIB modules, you must concatenate the various subfiles before the **snmpinfo** command can successfully use the resultant **mib.defs** file. The SMI subfile must be at the top of the final objects definition file.

You can add objects definitions for experimental MIB modules or private-enterprise-specific MIB modules to the **/etc/mib.defs** file, but you must first obtain the private MIB module from the vendor that supports those MIB variables.

To update the **/etc/mib.defs** file to incorporate a vendor's private or experimental MIB objects definitions, create a subfile and then concatenate that subfile to the existing MIB II **/etc/mib.defs** file. See example 3.

Flags

-o *OutputFile*	Defines the path and file name of the MIB objects definition file for the **snmpinfo** command. There is no default path and file name for this flag. If this flag is not specified, the objects definition file is not created.
-s	Suppresses the conversion verification messages. If this flag is not specified, the conversion verification messages are printed to standard output.

Parameters

InputFile	Defines the ASN.1 object definitions module for input to the **mosy** compiler. This file can be formatted according to either the **smi.my** or **mibll.my** file format.

Examples

1. To create an objects definition file for use by the **snmpinfo** command with one pass of the **mosy** command, enter:

   ```
   mosy -o /etc/mib.defs /usr/samples/snmpd/smi.my
   /usr/samples/snmpd/mibII.my
   ```

   In this example, **/usr/samples/snmpd/smi.my** and **/usr/samples/snmpd/mibll.my** are both specified as input files and the resultant objects definition file is the **/etc/mib.defs** file.

2. To create objects definition subfiles, enter:

   ```
   mosy -o /tmp/smi.obj /usr/samples/snmpd/smi.my
   mosy -o /tmp/mibII.obj /usr/samples/snmpd/mibII.my
   cat /tmp/smi.obj /tmp/mibII.obj > /etc/mib.defs
   ```

In this example, the first command creates an SMI objects file, /tmp/smi.obj, from the **/usr/samples/snmpd/smi.my** file. The second command creates the MIB objects definition file, /tmp/mibII.obj, from the **/usr/samples/snmpd/mibII.my** file. The final command concatenates the subfiles, placing the SMI objects definition file first in the resultant **/etc/mib.defs** file.

3. To add private enterprise specific MIB objects definitions to an existing **/etc/mib.defs** file for use by the **snmpinfo** command, enter:

```
mosy -o /tmp/private.obj /tmp/private.my
cat /etc/mib.defs /tmp/private.obj > /tmp/mib.defs
mv /tmp/mib.defs /etc/mib.defs
```

In this example, the first command creates the /tmp/private.obj objects definition file. The second command concatenates the **/etc/mib.defs** MIB objects definition file with the /tmp/private.obj private MIB file and places the concatenated contents into the /tmp/mib.defs temporary MIB objects definition file. The final command moves the temporary file to the **/etc/mib.defs** file for use by the **snmpinfo** command.

Files

/etc/mib.defs	Defines the Management Information Base (MIB) variables the SNMP agent should recognize and handle. The format of the **/etc/mib.defs** file is required by the **snmpinfo** command.
/usr/samples/snmpd/smi.my	Defines the ASN.1 definitions by which the SMI is defined as in RFC 1155.
/usr/samples/snmpd/mibII.my	Defines the ASN.1 definitions for the MIB II variables as defined in RFC 1213.

Related Information

The **snmpinfo** command.

Understanding the Management Information Base (MIB), Understanding Terminology Related to Management Information Base (MIB) Variables in *AIX 5L Version 5.2 Communications Programming Concepts*.

---

mount Command

Purpose

Makes a file system available for use.

Syntax

mount [**-f**] [**-n** *Node*] [**-o** *Options*] [**-p**] [**-r**] [**-v** *VfsName*] [**-t** *Type* | [*Device* | *Node***:***Directory*] *Directory* | **all** | **-a**] [**-V** [*generic_options*] *special_mount_points*]

Description

The **mount** command instructs the operating system to make a file system available for use at a specified location (the mount point). In addition, you can use the **mount** command to build other file trees made up of directory and file mounts. The **mount** command mounts a file system expressed as a device using the *Device* or *Node***:***Directory* parameter on the directory specified by the *Directory* parameter. After the **mount** command has finished, the directory specified becomes the root directory of the newly mounted file system.

Only users with root authority or are members of the system group and have write access to the mount point can issue file or directory mounts. The file or directory may be a symbolic link. The **mount** command uses the real user ID, not the effective user ID, to determine if the user has appropriate access. System

group members can issue device mounts, provided they have write access to the mount point and those mounts specified in the **/etc/filesystems** file. Users with root user authority can issue any **mount** command.

Users can mount a device provided they belong to the system group and have appropriate access. When mounting a device, the **mount** command uses the *Device* parameter as the name of the block device and the *Directory* parameter as the directory on which to mount the file system.

If you enter the **mount** command without flags, the command displays the following information for the mounted file systems:

- the node (if the mount is remote)
- the object mounted
- the mount point
- the virtual-file-system type
- the time mounted
- any mount options

If you specify only the *Directory* parameter, the **mount** command takes it to be the name of the directory or file on which a file system, directory, or file is usually mounted (as defined in the **/etc/filesystems** file). The **mount** command looks up the associated device, directory, or file and mounts it. This is the most convenient way of using the **mount** command, because it does not require you to remember what is normally mounted on a directory or file. You can also specify only the device. In this case, the command obtains the mount point from the **/etc/filesystems** file.

The **/etc/filesystems** file should include a stanza for each mountable file system, directory, or file. This stanza should specify at least the name of the file system and either the device on which it resides or the directory name. If the stanza includes a mount attribute, the **mount** command uses the associated values. It recognizes five values for the mount attributes: **automatic**, **true**, **false**, **removable**, and **readonly**.

The **mount all** command causes all file systems with the **mount=true** attribute to be mounted in their normal places. This command is typically used during system initialization, and the corresponding mounts are referred to as automatic mounts.

The **mount** command can also be used to access a snapshot of a JFS2 file system as a directory tree. The snapshot on *Device* is mounted read-only at *Directory*. A snapshot can only be mounted once. When mounting a JFS2 file system with snapshots, the snapshots are activated.

You can use the File Systems application in Web-based System Manager (wsm) to run this command. You could also use the System Management Interface Tool (SMIT) **smit mount** fast path to run this command.

Note: If the **mount** command encounters a Journaled File System (JFS) or Enhanced Journaled File System (JFS2) which was not unmounted before reboot, a replay of any JFS or JFS2 log records is attempted. In order to move a compatible JFS filesystem to a system running an earlier release of the operating system, the filesystem must always be unmounted cleanly prior to its movement. Failure to unmount first may result in an incompatible JFS log device. If the movement results in an unknown log device, the filesystem should be returned to the system running the latter operating system release, and **fsck** should be run on the filesystem.

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	Mounts all file systems in the **/etc/filesystems** file with stanzas that contain the **true** mount attribute.
all	Same as the **-a** flag.
-f	Requests a forced mount during system initialization to enable mounting over the root file system.
-n *Node*	Specifies the remote node that holds the directory to be mounted.

File System Specific Options

-o *Options*

Specifies options. Options entered on the command line should be separated only by a comma. The following file system-specific options do not apply to all virtual file system types:

bsy Prevents the mount operation if the directory to be mounted over is the current working directory of a process.

cio Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this filesystem will behave as if they had been opened with **O_CIO** specified in the **open()** system call.

dio Specifies that I/O on the filesystem will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call.

fmode=*octal*
Specifies the mode for a file and directory. The default is 755.

gid=gid
Specifies the GID that is assigned to files in the mount. The default is **bin**.

log=*LVName*
Specifies the full path name of the filesystem logging logical volume name where the following file-system operations are logged.

nocase
Turns-off case mapping. This is useful for cdroms using the ISO 9660:1998/HSG standard.

nodev Specifies that you cannot open devices from this mount. This option returns a value of **ENXIO** if a failure occurs.

nosuid Specifies that execution of **setuid** and **setgid** programs by way of this mount is not allowed. This option returns a value of **EPERM** if a failure occurs.

rbr Mount filesystem with the release-behind-when-reading capability. When sequential reading of a file in this filesystem is detected, the real memory pages used by the file will be released once the pages are copied to internal buffers.
Note: When **rbr** is specified, the **D_RB_READ** flag is ultimately set in the **_devflags** field in the **pdtentry** structure.

rbw Mount filesystem with the release-behind-when-writing capability. When sequential writing of a file in this filesystem is detected, the real memory pages used by the file will be released once the pages written to disk.
Note: When **rbw** is specified, the **D_RB_WRITE** flag is set.

rbrw Mount filesystem with both release-behind-when-reading and release-behind-when-writing capabilities.
Note: If **rbrw** is specified, both the **D_RB_READ** and the **D_RB_WRITE** flags are set.

ro Specifies that the mounted file is read-only. The default value is **rw**.

rw Specifies that the mounted file is read/write accessible. **rw** is the default value.

snapshot
Specifies the *Device* to be mounted is a snapshot. The snapped file system for the specified snapshot must already be mounted or an error message will display.

snapto=*snapshot*
Specifies the location to start a snapshot with the value of *snapshot* when mounting the specified JFS2 file system.

File System Specific Options (continued)

-o *Options* Specifies options. Options entered on the command line should be separated only by a comma. The following file system-specific options do not apply to all virtual file system types:

upcase
Changes case mapping from default lowercase to uppercase. This is useful for cdroms using the ISO 9660:1998/HSG standard.

uid=uid
Specifies the UID that is assigned to files in the mount, the defeult is bin.

wrkgrp=workgroup
Specifies the workgroup that the SMB server belongs.

NFS Specific Options

-o *Options* Specifies options. Options you enter on the command line should be separated only by a comma, not a comma and a space. The following NFS-specific options do not apply to all virtual file system types:

acdirmax=*n*
Holds cached attributes for no more than *n* seconds after directory update. The default is 60 seconds.

acdirmin=*n*
Holds cached attributes for at least *n* seconds after directory update. The default is 30 seconds.

acl Requests using the Access Control List RPC program for this NFS mount. If the **acl** option is used, the ACL RPC program is used only if the NFS server provides it. The default is **noacl**. This option only applies to AIX 4.2.1 or later.

acregmax=*n*
Holds cached attributes for no longer that *n* seconds after file modification. The default is 60 seconds.

acregmin=*n*
Holds cached attributes for at least *n* seconds after file modification. The default is 3 seconds.

actimeo=*n*
Sets minimum and maximum times for regular files and directories to *n* seconds. If this option is set, it overrides any settings for the **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** options.

bg Attempts mount in background if first attempt is unsuccessful. The default value is **fg**.

fastattr Bypasses the requirement that files currently being written will be sent to the server before the attributes of the file is read. This option is to be used with caution, since it will cause the client to assume that the file data that has not yet reached the server will be written without problem. In case of write errors, the client and server will have different opinions on what the size of the file really is. Likewise, a client will not be aware of attribute changes to the file being made by another client, so this option must not be used in environments where two clients are writing to the same files.

fg Attempts mount in foreground if first attempt is unsuccessful. **fg** is the default value.

grpid Directs any file or directory created on the file system to inherit the group ID of the parent directory.

hard Retries a request until server responds. The option is the default value.

intr Allows keyboard interrupts on hard mounts.

llock Requests that files lock locally at the NFS client. NFS network file locking requests are not sent to the NFS server if the **llock** option is used. This option only applies to AIX 4.2.1 or later.

maxgroups=*n*
Indicates that NFS RPC calls using **AUTH_UNIX** may include up to *n* member groups of information. Using this option to increase the number of member groups beyond the RPC protocol standard of 16 will only work against servers that support more than 16 member groups. Otherwise, the client will experience errors.

Values below 16 or greater than 64 will be ignored. By default, the protocol standard maximum of 16 is adhered to. AIX NFS servers will accept and process **AUTH_UNIX** credentials with up to 64 groups starting with AIX 5L Version 5.2 with the 5200-01 Recommended Maintenance package. The actual number of member groups sent by the NFS client is dependent on the number of groups the involved user is a member of, and may be limited by the length of the NFS client's hostname (which is included in the **AUTH_UNIX** information).

noac Specifies that the **mount** command performs no attribute or directory caching. If you do not specify this option, the attributes (including permissions, size, and timestamps) for files and directories are cached to reduce the need to perform over-the-wire **NFSPROC_GETATTR** Remote Procedure Calls (RPCs). The **NFSPROC_GETATTR** RPC enables a client to prompt the server for file and directory attributes. The **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** options control the length of time for which the cached values are retained.

noacl Specifies not to use the Access Control List RPC program for this NFS mount request. The default is **noacl**. This option only applies to AIX 4.2.1 or later.

nointr Specifies no keyboard interrupts allowed on hard mounts. This option only applies to AIX 4.2.1 or later.

port=*n* Sets server Internet Protocol (IP) port number to n. The default value is the 2049.

posix Requests that pathconf information be exchanged and made available on an NFS Version 2 mount. Requires a mount Version 2 **rpc.mountd** at the NFS server.

This option only applies to AIX 4.2.1 or later.

proto=[udp|tcp]
Specifies the transport protocol. The default transport protocol used for AIX 4.2.1 NFS mounts is **udp** if available on the server. For AIX 4.3 and later the default is **tcp**. Use the **proto=[udp|tcp]** option to override the default. This option only applies to AIX 4.2.1 or later.

retrans=*n*
Sets the number of NFS transmissions to *n*. The default value is 5.

retry=*n*
Sets the number of times the mount is attempted to *n*; the default value is 1000. When the retry value is 0, the system makes 10,000 attempts.

rsize=*n*
Sets the read buffer size to *n* bytes. The default value is 8192. For AIX 4.2.1 only, the default value is 32768 when using Version 3 of the NFS protocol.

secure Specifies that the **mount** command uses Data Encryption Standard (DES) for NFS transactions.

shortdev
Specifies that you are mounting a file system from a host that does not support 32-bit device special files. This option does not apply to AIX 4.2.1 or later.

soft Returns an error if the server does not respond. The default value is **hard**.

timeo=*n*

Sets the Network File System (NFS) time-out period to *n* tenths of a second. For TCP mounts, default timeout is 100, which equals 10 seconds. For UDP mounts, default timeout is 11, which equals 1.1 seconds, but varies depending on the NFS operation taking place. For UDP mounts, the timeout will increase for each failed transmission, with a maximum value of 20 seconds. Each transmission will be attempted twice, after which the time-out value is updated.

vers=[2|3]

Specifies NFS version. The default is the version of NFS protocol used between the client and server and is the highest one available on both systems. If the NFS server does not support NFS Version 3, the NFS mount will use NFS Version 2. Use the **vers=[2|3]** option to select the NFS version. This option only applies to AIX 4.2.1 or later.

wsize=*n*

Sets the write buffer size to *n* bytes. The default value is 8192. For AIX 4.2.1 only, the default value is 32768 when using Version 3 of the NFS protocol.

-p Mounts a file system as a removable file system. While open files are on it, a removable mounted file system behaves the same as a normally mounted file system. However, when no files are open (and no process has a current directory on the file system), all of the file system disk buffers in the file system are written to the medium, and the operating system forgets the structure of the file system.

-r Mounts a file system as a read-only file system, regardless of its previous specification in the **/etc/filesystems** file.

-t *Type* Mounts all stanzas in the **/etc/filesystems** file that contain the **type=***Type* attribute and are not mounted. The *Type* parameter specifies the name of the group.

-v *VfsName* Specifies that the file system is defined by the *VfsName* parameter in the **/etc/vfs** file.

CacheFS Specific Options

The CacheFS-specific version of the **mount** command mounts a cached file system; if necessary, it NFS-mounts its back file system. It also provides a number of CacheFS-specific options for controlling the caching process.

To mount a CacheFS file system, use the **mount** command with the **-V** flag followed by the argument. The following **mount** flags are available:

The following arguments to the **-o** flag are specifically for CacheFS mounts. Options you enter on the command line should be separated only by a comma, not a comma and a space.

Note: The **backfstype** argument must be specified.

-o Specifies options.

acdirmin=*n*

Specifies that cached attributes are held for at least *n* seconds after directory update. After *n* seconds, CacheFS checks to see if the directory modification time on the back file system has changed. If it has, all information about the directory is purged from the cache and new data is retrieved from the back file system. The default value is 30 seconds.

acregmax=*n*

Specifies that cached attributes are held for no more than n seconds after file modification. After *n* seconds, all file information is purged from the cache. The default value is 30 seconds.

acregmin=*n*
Specifies that cached attributes are held for at least *n* seconds after file modification. After *n* seconds, CacheFS checks to see if the file modification time on the back file system has changed. If it has, all information about the file is purged from the cache and new data is retrieved from the back file system. The default value is 30 seconds.

actimeo=*n*
Sets **acregmin**, **acregmax**, **acdirmin**, and **acdirmax** to *n*.

backfstype=*file_system_type*
The file system type of the back file system (for example, nfs).

backpath=*path*
Specifies where the back file system is already mounted. If this argument is not supplied, CacheFS determines a mount point for the back file system. The back file system must be read-only.

cachedir=*directory*
The name of the cache directory.

cacheid=*ID*
ID is a string specifying a particular instance of a cache. If you do not specify a cache ID, CacheFS will construct one.

demandconst
Enables maximum cache consistency checking. By default, periodic consistency checking is enabled. When you enable **demandconst**, it checks on every read and write.

local_access
Causes the front file system to interpret the mode bits used for access checking instead or having the back file system verify access permissions. Do not use this argument with secure NFS.

noconst
Disables cache consistency checking. By default, periodic consistency checking is enabled. Specify **noconst** only when you know that the back file system will not be modified. Trying to perform cache consistency check using **cfsadmin-s** will result in error. **demandconst** and **noconst** are mutually exclusive.

purge Purge any cached information for the specified file system.

rw | ro Read-write (default) or read-only.

suid | nosuid
Allow (default) or disallow set-uid execution

write-around | non-shared
Write modes for CacheFS. The write-around mode (the default) handles writes the same as NFS does; that is, writes are made to the back file system, and the affected file is purged from the cache. You can use the non-shared mode when you are sure that no one else will be writing to the cached file system.

-V Mounts a CacheFS file system.

Examples

1. To list the mounted file systems, enter:

```
mount
```

This command produces output similar to the following:

```
node   mounted          mounted    vfs  date          options   over
----   -------          ---------  ---  ------------   -------  ---------
       /dev/hd0         /          jfs  Dec 17 08:04   rw, log  =/dev/hd8
       /dev/hd3         /tmp       jfs  Dec 17 08:04   rw, log  =/dev/hd8
       /dev/hd1         /home      jfs  Dec 17 08:06   rw, log  =/dev/hd8
       /dev/hd2         /usr       jfs  Dec 17 08:06   rw, log  =/dev/hd8
sue    /home/local/src  /usr/code  nfs  Dec 17 08:06   ro, log  =/dev/hd8
```

For each file system, the **mount** command lists the node name, the device name, the name under which it is mounted, the virtual-file-system type, the date and time it was mounted, and its options.

2. To mount all default file systems, enter:

   ```
   mount  all
   ```

   This command sequence mounts all standard file systems in the **/etc/filesystems** file marked by the **mount=true** attribute.

3. To mount a remote directory, enter:

   ```
   mount  -n nodeA /home/tom.remote /home/tom.local
   ```

   This command sequence mounts the `/home/tom.remote` directory located on `nodeA` onto the local `/home/tom.local` directory. It assumes the default *VfsName* parameter=**remote**, which must be defined in the **/etc/vfs** file.

4. To mount a file or directory from the **/etc/filesystems** file with a specific type, enter:

   ```
   mount  -t remote
   ```

   This command sequence mounts all files or directories in the **/etc/filesystems** file that have a stanza that contains the **type=remote** attribute.

5. To CacheFS-mount the file system which is already NFS-mounted on **/usr/abc**, enter:

   ```
   mount -V cachefs -o backfstype=nfs,backpath=/usr/abc,
   cachedir=/cache1 server1:/user2 /xyz
   ```

   The lines similar to the following appear in the **/etc/mnttab** file after the mount command is executed:

   ```
   server1:/user2 /usr/abc nfs
   /usr/abc /cache1/xyz cachefs backfstype=nfs
   ```

6. To mount a snapshot, type:

   ```
   mount -o snapshot /dev/snapsb /home/janet/snapsb
   ```

   This command mounts the snapshot contained on the **/dev/snapsb** device onto the **/home/janet/snapsb** directory.

7. To mount a file system and create a snapshot, type:

   ```
   mount -o snapto=/dev/snapsb /dev/sb /home/janet/sb
   ```

   This command mounts the file system contained on the **/dev/sb**device onto the **/home/janet/sb** directory and creates a snapshot for the file system on the **/dev/snapsb**device.

8. To access files on an SMB server as a local filesystem, type:

   ```
   mount -v cifs -n pezman/user1/pass1 -o uid=201,fmode=750 /home /mnt
   ```

Files

/etc/filesystems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual-file-system types.

Related Information

The **cdcheck** command, **cdeject** command, **cdmount** command, **cdromd** command, **cdumount** command, **cdutil** command, **nfso** command, **umount** command.

The **mntctl** subroutine, **mount** subroutine, **umount** subroutine.

The **filesystems** file, **vfs** file.

For information on installing the Web-based System Manager, see Chapter 2: Installing Web-based System Manager in *AIX 5L Version 5.2 Web-based System Manager Administration Guide*.

Mounting Overview and System Management Interface Tool (SMIT) in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

Performance Implications of Hard or Soft NFS Mounts and Other mount Options That Affect Performance in *AIX 5L Version 5.2 Performance Management Guide*.

---

mountd Daemon

Purpose

Answers requests from clients for file system mounts.

Syntax

/usr/sbin/rpc.mountd [**-n**] [**-N**]

Description

The **mountd** daemon is a Remote Procedure Call (RPC) that answers a client request to mount a file system. The **mountd** daemon finds out which file systems are available by reading the **/etc/xtab** file.

In addition, the **mountd** daemon provides a list of currently mounted file systems and the clients on which they are mounted. You can display this list by using the **showmount** command.

The **mountd** daemon listens for requests on the port specified in the **/etc/services** file for the service **mountd**. If the **/etc/services** file does not specify a port, one will be chosen when the daemon starts. For example, adding the lines:

```
mountd   6666/tcp
mountd   6666/udp
```

will cause **mountd** to listen for requests on port 6666.

Examples

The **mountd** daemon is started from the **/etc/rc.nfs** file. The **mountd** daemon can be started and stopped by the following System Resource Controller (SRC) commands:

```
startsrc -s rpc.mountd
stopsrc -s rpc.mountd
```

To change the parameters passed to the **mountd** daemon, use the **chssys** command. For example:

```
chssys -s rpc.mountd -a Argument
```

The change will not take effect until the daemon is restarted.

Flags

-n Allows clients that use older versions of NFS to mount file systems. This option makes the system less secure. It is the default.

-N Deny mount requests originating from non-privileged ports. This is the opposite of using the **-n** flag, and is not enabled by default.

Files

/etc/exports	Lists the directories that the server can export.
/etc/inetd.conf	Defines how the **inetd** daemon handles Internet service requests.
/etc/xtab	Lists currently exported directories.
/etc/services	Defines the sockets and protocols used for Internet services. Contains information about the known services used in the DARPA Internet network.

Related Information

The **chssys** command, **mount** command, **showmount** command.

The **nfsd** daemon, **portmap** daemon.

Network File System (NFS) Overview for System Management in *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

System Resource Controller Overview in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

How to Mount a NFS File System Explicitly and How to Mount a File System Using Secure NFS in *AIX 5L Version 5.2 Security Guide*.

List of NFS Commands.

---

mpcfg Command

Purpose

Manages remote maintenance service information.

Syntax

Display Service Information

mpcfg -d { **-f -m -p -S**}

Change Service Information

mpcfg -c { **-f** | **-m** | **-p -S -w***} Index Value*...

Save or Restore Service Information

mpcfg { **-r** | **-s** }

Description

The **mpcfg** command enables a user with root authority to manage service information consisting of the service support and diagnostic flags (**-S** and **-f** flags), the modem and site configuration (**-m** flag), and the remote support phone numbers (**-p** flag).

The **mpcfg** command 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 Series for AIX 5.1 and earlier.

You can perform the following actions:

- Display (**-d** flag) the service information
- Change (**-c** flag) the service information
- Save (**-s** flag) the service information in the **/etc/lpp/diagnostics/data/bump** file
- Restore (**-r** flag) the service information to values read in the **/etc/lpp/diagnostics/data/bump** file.

Note: Generally the **mpcfg** command is not used directly but through the **diag** command.

Flags

-c Changes values of service information. The values that you want to modify are identified first by the flag **-f**, **-m**, **-p** or **-S**, and then by their index (*Index* parameter*)* within this category. The new value to assign (*Value* parameter*)* directly follows this index in the command. You can specify only one flag among **-f**, **-m**, **-p**, **-S** but several pairs *"Index Value"*. The flag **-w** allows to change maintenance passwords.

-d Displays the values of service information according to the **-f**, **-m**, **-p** and **-S** flags set in the command. These values are displayed associated with their corresponding indexes and names.

-r Reads the service information in the **/etc/lpp/diagnostics/data/bump** file, and restores it in the non volatile memory (NVRAM).

-s Saves the service information in the **/etc/lpp/diagnostics/data/bump** file.

-f Indicates that the action (display or change) will be applied to the diagnostic flags.

-m Indicates that the action (display or change) will be applied to the modem and site configuration.

-p Indicates that the action (display or change) will be applied to the remote support phone numbers.

-S Indicates that the action (display or change) will be applied to the service support flags.

-w Indicates that the change will be applied to a password.

Security

Access Control: only the root user can run this command.

Examples

1. To display the modem and site configuration, enter the following command:

   ```
   mpcfg  -d m
   ```

   This produces output similar to the following:

   ```
   Index   Name                            Value
   1       Modem Parameters File Name      /usr/share/modems/plextel
   2       Service Line Speed              2400
   3       Protocol Inter Data Block Delay 15
   4       Protocol Time Out               30
   5       Retry Number                    2
   6       Customer ID                     xyz
   7       Login ID                        abcd
   8       Password ID                     %qw!as
   ```

2. To assign the new value `22114433` to the first remote support phone number, enter the following command:

   ```
   >mpcfg  -c  -p 1 22114433
   ```

3. To save the service information in the **/etc/lpp/diagnostics/data/bump** file, enter the following command:

```
mpcfg -s
```

4. To restore the service information from the **/etc/lpp/diagnostics/data/bump** file to NVRAM, enter the following command:

```
mpcfg -r
```

Files

/usr/sbin/mpcfg	Contains the **mpcfg** command.
/etc/lpp/diagnostics/data/bump	Contains the service support and diagnostic flags, remote support phone numbers, and modem and site configuration.

mpcstat Command

Purpose

Displays operational information about a Multi-Protocol Over ATM (MPOA) Client.

Syntax

mpcstat [**-a -c -e -i -m -r -s -t -v**] [*Device_Name*]

Description

This command displays Multi-Protocol Over ATM (MPOA) Client operational information gathered by a specified MPOA Client device. If a MPOA Client (MPC) device name is not entered, information for the available MPC appear. You can use the flags to narrow down your search to specify specific categories of information such as Configuration, Egress Cache Entries, Ingress Cache Entries, MPOA Servers, Shortcut Virtual Connections, and Statistics, or you can elect to have all of the information categories display.

You can also toggle debug tracing on or off and reset statistics counters.

Parameters

Device_Name	The name of the MPOA Client device name, for example, *mpc0*.

Flags

-a	Requests that all of the MPOA Client information appear. Note that this flag does not reset statistics counters or toggle trace. If a flag is not entered, the **-a** flag is the default flag.
-c	Requests the configuration
-e	Requests the egress (incoming) cache
-i	Requests the ingress (outgoing) cache.
-m	Requests the list of MPOA Servers in use.
-r	Resets the statistics counters after reading.
-s	Requests the statistics counters.
-t	Toggles full debug trace on or off.
-v	Requests the list of Shortcut Virtual Connections.

The following information appears for all valid calls and contains the following fields:

Device Name
Displays the device name of the MPOA Client.

MPC State
Displays the current state of the MPOA Client.

Example States:

```
Idle              Registering with the ELAN.
Initializing      Registering with the switch.
Operational       Fully operational.
Network Down      Network is currently unavailable.
```

MPC Address
Displays the MPOA Client's 20-byte ATM address for a specific ATM adapter port device name. The adapter port device name is also displayed.

Elapsed Time
Displays the real time period which has elapsed since statistics were last reset.

MPC Configuration
Selected with the **-a** or **-c** flags. Displays the network administrator's pre-configured attributes for the MPOA Client, or the values provided by a Lan Emulation Configuration Server (LECS).

MPC Egress Cache
Selected with the **-a** or **-e** flags. Displays the current egress cache entries. Included are the state of the entry, its Level-3 address, and ATM shortcut address, as well as additional descriptive values associated with each entry.

Example States

```
Active         Has active shortcut connection.
Purging        Purging the egress MPOA Server entry.
DP Purge       Purging the remote MPOA Client data plane.
Inactive       No current activity on shortcut connection.
```

MPC Ingress Cache:
Selected with the **-a** or **-i** flags. Displays the current ingress cache entries. Included are the state of the entry, its Level-3 address, and ATM shortcut address, as well as additional descriptive values associated with each entry.

Example States

```
Flow Detect    Waiting for packet threshold to enable shortcut.
Resolution     Packet threshold reached, resolving shortcut.
Hold Down      Shortcut resolution failed, waiting for retry.
Resolved       Shortcut resolution sequence complete.
```

MPOA Server List
Selected with the **-a** or **-m** flags. Displays a list of MPOA Servers currently known by this MPC. Included in each entry are the name of the LE Client that identified the MPS, the MPS ATM address, and the MPS LAN MAC address.

MPC Statistics
Selected with the **-a** or **-s** flags. Displays the current Transmit, Receive, and General statistics for this MPOA Client.

Shortcut Virtual Connection
Selected with the **-a** or **-v** flags. Displays the current list of shortcut virtual circuits in use by the MPOA client. Included are virtual path and channel values, VC state, ATM device name, as well as additional descriptive values associated with each entry.

Example States:

```
Idle            Call idle.
Signalling      Call placed but not established.
Operational     Call connected; data path valid.
Released        Call released.
Retry           Temporary call failure; will be retried.
Hold Down       Call failure; will be suspended for hold down period.
```

Exit Status

If an invalid *Device_Name* is specified, this command produces error messages stating that it could not connect to the device. Note that MPOA is a protocol extension to the ATM LAN Emulation protocol, and must have a corresponding and available LE Client to be operational. Examples of an invalid device error message are:

```
MPCSTAT: Device is not an MPOA device.
MPCSTAT: No LEC device with MPOA enabled.
MPCSTAT: Device is not available.
```

Related Information

The **atmstat** command, **entstat** command, **lecstat** command, and **tokstat** command.

---

mrouted Daemon

Purpose

Forwards a multicast datagram. This daemon only applies to AIX 4.2.1 or later.

Syntax

/usr/sbin/mrouted [**-p**] [**-c** *Config_File*] [**-d** [*Debug_Level*]]

Description

The **mrouted** daemon is an implementation of the Distance Vector Multicast Routing Protocol (DVMRP), an earlier version of which is specified in RFC 1075. It maintains topological knowledge using a distance vector routing protocol (like RIP, described in RFC 1058), on which it implements a multicast datagram forwarding algorithm called Reverse Path Multicasting.

The **mrouted** daemon forwards a multicast datagram along a shortest (reverse) path tree rooted at the subnet on which the datagram originates. The multicast delivery tree may be thought of as a broadcast delivery tree that has been pruned back so that it does not extend beyond those subnetworks that have members of the destination group. Hence, datagrams are not forwarded along those branches that have no listeners of the multicast group. The IP time-to-live of a multicast datagram can be used to limit the range of multicast datagrams.

To support multicasting among subnets that are separated by (unicast) routers that do not support IP multicasting, the **mrouted** daemon includes support for tunnels, which are virtual point-to-point links between pairs of the **mrouted** daemons located anywhere in an Internet. IP multicast packets are encapsulated for transmission through tunnels, so that they look like typical unicast datagrams to intervening routers and subnets. The encapsulation is added on entry to a tunnel, and stripped off on exit from a tunnel. By default, the packets are encapsulated using the IP-in-IP protocol (IP protocol number 4). Older versions of the **mrouted** tunnel use IP source routing, which puts a heavy load on some types of routers. This version does not support IP source-route tunneling.

The tunneling mechanism allows the **mrouted** daemon to establish a virtual Internet, for the purpose of multicasting only, which is independent of the physical Internet and which may span multiple Autonomous Systems. This capability is intended for experimental support of Internet multicasting only, pending

widespread support for multicast routing by the regular (unicast) routers. The **mrouted** daemon suffers from the well-known scaling problems of any distance-vector routing protocol and does not support hierarchical multicast routing.

The **mrouted** daemon automatically configures itself to forward on all multicast-capable interfaces (that is, interfaces that have the IFF_MULTICAST flag set, excluding the loopback interface), and it finds other **mrouted** daemons directly reachable using those interfaces.

The **mrouted** daemon does not initiate execution if it has fewer than two enabled virtual interfaces, where a virtual interface (Vif) is either a physical multicast-capable interface or a tunnel. It logs a warning if all of its virtual interfaces are tunnels; such an **mrouted** daemon's configuration would be better replaced by more direct tunnels.

The **mrouted** daemon handles multicast routing only; there might be unicast-routing software running on the same machine as the **mrouted** daemon. With the use of tunnels, it is unnecessary for the **mrouted** daemon to have access to more than one physical subnet to perform multicast forwarding.

Flags

-c *Config_File* Starts the **mrouted** command using an alternate configuration file specified by the *Config_File* variable.

There are five types of configuration entries:

```
phyint local-addr [disable] [metric m] [threshold t] [rate_limit b]
[boundary (boundary-name|scoped-addr/mask-len)] [altnet
network/mask-len]

tunnel local-addr remote-addr
 [
metric m
] [
threshold t
] [
rate_limit b
]


[
boundary
 (
boundary-name
 |
scoped-addr
/
mask-len
)]


cache_lifetime ct
pruning off
|
on
name boundary-name scoped-addr
/
mask-len
```

See mrouted.conf File in *AIX 5L Version 5.2 Files Reference* for more information.

-d Sets the debug level. If no **-d** option is given, or if the debug level is specified as 0, the**mrouted** daemon detaches from the invoking terminal. Otherwise, it remains attached to the invoking terminal and responsive to signals from that terminal. If **-d** is given with no argument, the debug level defaults to 2. Regardless of the debug level, the **mrouted** daemon always writes warning and error messages to the system log demon. Non-zero debug levels have the following effects:

level 1 All syslog'ed messages are also printed to **stderr**.

level 2 All level 1 messages plus notifications of significant events are printed to **stderr**.

level 3 All level 2 messages plus notifications of all packet arrivals and departures are printed to **stderr**.

Upon startup, the **mrouted** daemon writes its pid to the file **/etc/mrouted.pid**.

-p Turns off pruning. Default is pruning enabled.

Signals

The following signals can be sent to the **mrouted** daemon:

HUP Restarts the **mrouted** daemon. The configuration file is reread every time this signal is evoked.
INT Terminates execution gracefully (that is, by sending good-bye messages to all neighboring routers).
TERM Same as **INT**.
USR1 Dumps the internal routing tables to **/usr/tmp/mrouted.dump**.
USR2 Dumps the internal cache tables to **/usr/tmp/mrouted.cache**.
QUIT Dumps the internal routing tables to **stderr** (if the **mrouted** daemon was invoked with a nonzero debug level).

For convenience in sending signals, the **mrouted** daemon writes its pid to **/etc/mrouted.pid** on startup.

Examples

1. To display routing table information, type:

```
kill -USR1 *cat /etc/mrouted.pid*
```

This produces the following output:

```
Virtual Interface Table
 Vif Local-Address                    Metric   Thresh   Flags
  0  36.2.0.8     subnet: 36.2           1        1     querier
                  groups: 224.0.2.1
                          224.0.0.4
                 pkts in: 3456
                pkts out: 2322323

  1  36.11.0.1    subnet: 36.11          1        1     querier
                  groups: 224.0.2.1
                          224.0.1.0
                          224.0.0.4
                 pkts in: 345
                pkts out: 3456

  2  36.2.0.8     tunnel: 36.8.0.77       3        1
                   peers: 36.8.0.77 (2.2)
              boundaries: 239.0.1
                        : 239.1.2
                 pkts in: 34545433
                pkts out: 234342

  3  36.2.0.8     tunnel: 36.6.8.23       3        16

Multicast Routing Table (1136 entries)
 Origin-Subnet    From-Gateway         Metric Tmr In-Vif Out-Vifs
```

```
36.2                                 1    45  0     1* 2 3*
36.8             36.8.0.77            4    15  2     0* 1* 3*
36.11                                1    20  1     0* 2 3*
.
.
.
```

In this example, there are four virtual interfaces connecting to two subnets and two tunnels. The Vif 3 tunnel is not in use (no peer address). The Vif 0 and Vif 1 subnets have some groups present; tunnels never have any groups. This instance of the **mrouted** daemon is the one responsible for sending periodic group membership queries on the Vif 0 and Vif 1 subnets, as indicated by the **querier** flags. The list of boundaries indicate the scoped addresses on that interface. A count of the no. of incoming and outgoing packets is also shown at each interface.

Associated with each subnet from which a multicast datagram can originate is the address of the previous hop router (unless the subnet is directly connected), the metric of the path back to the origin, the amount of time since an update for this subnet was last received, the incoming virtual interface for multicasts from that origin, and a list of outgoing virtual interfaces. The * (asterisk) means that the outgoing virtual interface is connected to a leaf of the broadcast tree rooted at the origin, and a multicast datagram from that origin will be forwarded on that outgoing virtual interface only if there are members of the destination group on that leaf.

The **mrouted** daemon also maintains a copy of the kernel forwarding cache table. Entries are created and deleted by the **mrouted** daemon.

2. To display cache table information, type:

```
kill -USR2 *cat /etc/mrouted.pid*
```

This produces the following output:

```
Multicast Routing Cache Table (147 entries)
 Origin         Mcast-group      CTmr   Age  Ptmr  IVif  Forwvifs
 13.2.116/22    224.2.127.255      3m    2m    -    0     1
 >13.2.116.19
 >13.2.116.196
 138.96.48/21   224.2.127.255      5m    2m    -    0     1
 >138.96.48.108
 128.9.160/20   224.2.127.255      3m    2m    -    0     1
 >128.9.160.45
 198.106.194/24 224.2.135.190      9m    28s   9m    0P
 >198.106.194.22
```

Each entry is characterized by the origin subnet number and mask and the destination multicast group. The **CTmr** field indicates the lifetime of the entry. The entry is deleted from the cache table when the timer decrements to zero. The Age field is the time since this cache entry was originally created. Because cache entries get refreshed if traffic is flowing, routing entries can grow very old. The **Ptmr** field is a hyphen if no prune was sent upstream or the amount of time until the upstream prune will time out. The **Ivif** field indicates the incoming virtual interface for multicast packets from that origin. Each router also maintains a record of the number of prunes received from neighboring routers for a particular source and group. If there are no members of a multicast group on any downward link of the multicast tree for a subnet, a prune message is sent to the upstream router. They are indicated by a P after the virtual interface number. The **Forwvifs** field shows the interfaces along which datagrams belonging to the source group are forwarded. A p indicates that no datagrams are being forwarded along that interface. An unlisted interface is a leaf subnet with are no members of the particular group on that subnet. A b on an interface indicates that it is a boundary interface, that is, traffic will not be forwarded on the scoped address on that interface. An additional line with a > (greater-than sign) as the first character is printed for each source on the subnet. There can be many sources in one subnet.

Files

/etc/mrouted.conf	Contains the configuration information for the **mrouted** daemon.
/usr/tmp/mrouted.dump	Contains the internal routing tables for the **mrouted** daemon.
/etc/mrouted.pid	Contains the process ID for the **mrouted** daemon.
/usr/tmp/mrouted.cache	Contains the internal cache tables for the **mrouted** daemon.

Related Information

/etc/**mrouted.conf** File in *AIX 5L Version 5.2 Files Reference*.

---

msgchk Command

Purpose

Checks for messages.

Syntax

msgchk [*User* ...]

Description

The **msgchk** command checks mail drops for messages. The **msgchk** command reports whether the mail drop for the specified user contains messages and indicates if the user has already seen these messages. By default, the **msgchk** command checks the mail drop for the current user.

Flags

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

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

Examples

1. To check to see if you have any new messages, enter:

   ```
   msgchk
   ```

   If you have new messages, the system responds with a message similar to the following:

   ```
   You have new Internet mail waiting
   ```

   If you have no messages, the system responds with a message similar to the following:

   ```
   You don't have any mail waiting
   ```

2. To check to see if user `karen` on your local system has any new messages, enter:

   ```
   msgchk karen
   ```

   In this example, if user `karen` on your local system has new messages, the system responds with a message similar to the following:

   ```
   karen has new Internet mail waiting
   ```

   If user `karen` on your local system has no messages, the system responds with a message similar to the following:

   ```
   karen doesn't have any mail waiting
   ```

Files

$HOME/.mh_profile	Contains the user's MH profile.
/etc/mh/mtstailor	Contains the MH tailor file.
/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/msgchk	Contains the **msgchk** command.

Related Information

The **inc** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

msh Command

Purpose

Creates a Message Handler (MH) shell.

Syntax

msh [*File*] [**-prompt** *String*] [**-notopcur** | **-topcur**]

Description

The **msh** command creates an MH shell for use with messages that are packed in a file. By default, this command looks for the **msgbox** file in the current directory. Within the MH shell, you can use the following MH commands:

```
ali         burst          comp        dist
folder         forw           inc         mark
mhmail         msgchk         next        packf
pick           prev           refile      repl
rmm            scan           send        show
sortm          whatnow        whom
```

These commands operate with limited functionality in the MH shell. To see how a command operates in the MH shell, enter the command name followed by the **-help** flag. Entering `help` or a `?` (question mark) displays a list of the MH commands you can use.

To leave the **msh** shell, press the Ctrl-D key sequence or enter `quit`.

Flags

-help	Lists the command syntax, available switches (toggles), and version information. **Note:** For MH, the name of this flag must be fully spelled out.
-notopcur	Makes the current message track the center line of the **vmh** scan window when the **msh** command is started using the **vmh** command. This flag is the default.
-prompt *String*	Prompts for the **msh** commands with the specified string.
-topcur	Makes the current message track the top line of the **vmh** scan window when the **msh** command is started using the **vmh** command.

Profile Entries

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

`fileproc:`	Specifies the program used to refile messages.
`Msg-Protect:`	Sets the protection level for your new message files.
`Path:`	Specifies the user's MH directory.
`showproc:`	Specifies the program used to show messages.

Examples

1. To start an **msh** shell, enter:

   ```
   msh
   ```

   If the **msgbox** file exists in the current directory, the system responds with a message similar to the following:

   ```
   Reading ./msgbox, currently at message 1 of 10
   ```

   Then, the system prompt appears as follows:

   ```
   (msh)
   ```

   In this example, the current message is message `1` in the `msgbox` file. You can now enter a modified subset of MH commands.
2. To start an **msh** shell to manipulate the messages stored in the `meetings` file, enter:

   ```
   msh meetings
   ```

Files

$HOME/.mh_profile	Specifies the user's MH profile.
/etc/mh/mtstailor	Contains the MH tailor file.
/usr/bin/msh	Contains the **msh** command.

Related Information

The **ali** command, **burst** command, **comp** command, **dist** command, **folder** command, **forw** command, **inc** command, **mark** command, **mhmail** command, **msgchk** command, **next** command, **packf** command, **pick** command, **prev** command, **refile** command, **repl** command, **rmm** command, **scan** command, **send** command, **show** command, **sortm** command, **vmh** command, **whatnow** command, **whom** command.

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

Mail Overview in *AIX 5L Version 5.2 System User's Guide: Communications and Networks*.

---

mt Command (BSD)

Purpose

Gives subcommands to streaming tape device.

Syntax

mt [**-f** *TapeName*] *Subcommand* [*Count*]

Description

The **mt** command gives subcommands to a streaming tape device. If you do not specify the **-f** flag with the *TapeName* parameter, the **TAPE** environment variable is used. If the environment variable does not exist, the **mt** command uses the **/dev/rmt0.1** device. The *TapeName* parameter must be a raw (not block) tape device. You can specify more than one operation with the *Count* parameter.

Subcommands

eof, weof	Writes the number of end-of-file markers specified by the *Count* parameter at the current position on the tape.
fsf	Moves the tape forward the number of files specified by the *Count* parameter and positions it to the beginning of the next file.
bsf	Moves the tape backwards the number of files specified by the *Count* parameter and positions it to the beginning of the last file skipped. If using the bsf subcommand would cause the tape head to move back past the beginning of the tape, then the tape will be rewound, and the **mt** command will return **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.
rewoff1, rewind	Rewinds the tape. The *Count* parameter is ignored.
status	Prints status information about the specified tape device. The output of the status command may change in future implementations.

Flag

-f *TapeName*	Specifies the *TapeName* parameter.

Examples

1. To rewind the `rmt1` tape device, enter:
   ```
   mt -f /dev/rmt1 rewind
   ```
2. To move forward two files on the default tape device, enter:
   ```
   mt fsf 2
   ```
3. To write two end-of-file markers on the tape in the `/dev/rmt0.6` file, enter:
   ```
   mt -f /dev/rmt0.6 weof 2
   ```

Exit Status

0	Indicates a successful completion.
>0	Indicates an error occurred.

Files

/dev/rmt/*n.n*	Specifies the raw streaming tape interface.
/usr/bin/mt	Contains the **mt** command file.

Related Information

The **tctl** command.

The **environment** file, **rmt** special file.

The **ioctl** subroutine.

Tape Drives in *AIX 5L Version 5.2 System Management Concepts: Operating System and Devices*.

mtrace Command

Purpose

Prints a multicast path from a source to a receiver.

Syntax

mtrace [**-l**] [**-M**] [**-n**] [**-p**] [**-s**] [**-U**] [**-g** *gateway*] [**-i** *if_addr*] [**-m** *max_hops*] [**-q** *nqueries*] [**-r** *resp_dest*] [**-S** *statint*] [**-t** *ttl*] [**-w** *wait*] *source* [*receiver*] [*group*]

Description

A trace query is passed hop-by-hop along the path from the receiver to the source, collecting hop addresses, packet counts, and routing error conditions along the path, and then the response is returned to the requestor. The default receiver is the host running the **mtrace** command, and the default group is 0.0.0.0.

Note: The **mtrace** command is intended for use in network testing, measurement, and management. Because the **mtrace** command heavily loads on the network, avoid using the **mtrace** command during typical operations or from automated scripts. It should be used primarily or with manual fault isolation. If the **-g** flag is specified, the source defaults to the host running **mtrace** and the receiver defaults to the router being addressed.

By default, the **mtrace** command first attempts to trace the full reverse path, unless the number of hops to trace is explicitly set with the **-m** flag. If there is no response within a 3-second timeout interval (changed with the **-w** flag), an * (asterisk) is printed and the probing switches to hop-by-hop mode. Trace queries are issued starting with a maximum hop count of one and increasing by one until the full path is traced or no response is received. At each hop, multiple probes are sent (default is three, changed with **-q** flag). The first half of the attempts (default is two) are made with the reply address set to standard multicast address, mtrace.mcast.net (224.0.1.32) with the *ttl* set to 32 more than what is needed to pass the thresholds seen so far along the path to the receiver. For each additional attempt, the *ttl* is increased by another 32 each time up to a maximum of 192. Because the desired router may not be able to send a multicast reply, the remainder of the attempts request that the response be sent via unicast to the host running the **mtrace** command.

Alternatively, the multicast *ttl* can be set explicitly with the **-t** flag, the initial multicast attempts can be forced to use unicast instead with the **-U** flag, the final unicast attempts can be forced to use multicast instead with the **-M** flag, or if you specify **-UM**, the **mtrace** command will first attempt using unicast and then multicast. For each attempt, if no response is received within the timeout, an * (asterisk) is printed. After the specified number of attempts have failed, the **mtrace** command will try to query the next hop router with a **DVMRP_ASK_NEIGHBORS2** request to see what kind of router it is. The **mtrace** command will try to query three (changed with the **-e** flag) hops past a non-responding router. Even though the **mtrace** command is incapable of sending a response, it might be capable of forwarding the request.

Flags

-g *gateway* Sends the trace query via unicast directly to the multicast router *gateway* rather than multicasting the query. This must be the last-hop router on the path from the intended source to the receiver.

-i *if_addr* Uses *if_addr* as the local interface address (on a multi-homed host) for sending the trace query and as the default for the receiver and the response destination.

-l Loops indefinitely printing packet rate and loss statistics for the multicast path every 10 seconds (see -S *stat_int*).

-m *max_hops* Sets *max_hops* to the maximum number of hops that will be traced from the receiver to the source. The default is 32 hops and infinity for the DVMRP routing protocol).

-M Always requests the response using multicast rather than attempting unicast for the last half of the tries.

-n Prints hop addresses numerically rather than symbolically and numerically (saves a name server address-to-name lookup for each router found on the path).

-p Listens passively for multicast responses from traces initiated by others. This works best when run on a multicast router.

-q *nqueries* Sets the maximum number of query attempts for any hop to *nqueries*. The default is 3.

-r *resp_dest* Sends the trace response to dhost rather than to the host on which the **mtrace** command is being run, or to a multicast address other than the one registered for this purpose (224.0.1.32).

-s Prints a short form output including only the multicast path and not the packet rate and loss statistics.

-S *statint* Changes the interval between statistics gathering traces to *statint* seconds (default 10 seconds).

-t *ttl* Sets the *ttl* (time-to-live, or number of hops) for multicast trace queries and responses. The default is 127, except for local queries to the `all routers` multicast group that use the *ttl* of 1.

-U Forces initial multicast attempts to use unicast instead.

-w *wait* Sets the time to wait for a trace response to*wait* seconds (default 3 seconds).

Parameters

source Specifies the host for which the multicast path from a particular receiver is sought. This is a required parameter.

receiver Specifies the host from which the multicast path is sought for a particular source. Default is the host in which the **mtrace** command is running. This is an optional parameter.

group Specifies the multicast group. This is an optional parameter.

Examples

In the following example, the two machines, 10.27.41.57 and 10.27.40.20, are on two different subnets separated by a router having two interfaces, 10.27.40.11 and 10.27.41.11. To find the multicast path from 10.27.40.20 to 10.27.41.57, type the following:

```
# mtrace 10.27.41.57 224.2.0.1
```

The following is displayed:

```
Mtrace from 10.27.41.57 to 10.27.40.20 via group 224.2.0.1
Querying full reverse path...
  0  ? (10.27.40.20)
 -1  ? (10.27.40.11)  DVMRP  thresh^ 1
 -2  ? (10.27.41.57)
Round trip time 1 ms; total ttl of 2 required.

Waiting to accumulate statistics... Results after 10 seconds:

  Source        Response Dest    Overall     Packet Statistics For Traffic From
10.27.41.57     224.0.1.32       Packet      10.27.41.57 To 224.2.0.1
     v       __/  rtt    1 ms     Rate       Lost/Sent = Pct  Rate
10.27.41.11
10.27.40.11     ?
     v      \__   ttl    2         0 pps
10.27.40.20     10.27.40.20
  Receiver      Query Source
```

Related Information

Related commands include the **mrouted** daemon and the **traceroute** command.

---

mv Command

Purpose

Moves files.

Syntax

To Move Files to a Directory Maintaining Original File Names

mv [**-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] *SourceFile TargetFile*

To Move and Rename a File or Directory

mv -E{**force**|**ignore**|**warn**}] [**-i** | **-f**] *SourceFile ... TargetDirectory*

Description

> **Attention:** The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

The **mv** command moves files and directories from one directory to another or renames a file or directory. If you move a file or directory to a new directory, it retains the base file name. When you move a file, all links to other files remain intact, except when you move it to a different file system. When you move a directory into an existing directory, the directory and its contents are added under the existing directory.

When you use the **mv** command to rename a file or directory, the *TargetDirectory* parameter can specify either a new file name or a new directory path name.

If moving the file would overwrite an existing file that does not have write-permission set and if standard input is a workstation, the **mv** command displays the file-permission code and reads a line from standard input. If that line begins with a y or the locale's equivalent of a y, the **mv** command moves the file. If the response is anything other than a y, the **mv** command does nothing to that file and continues with the next specified file. The file-permission code displayed may not fully represent the access permission if the *TargetFile* is associated with an ACL. When the parent directory of the *SourceFile* is writable and has the sticky bit set, one or more of the following conditions are true:

- The user must own the file.
- The user must own the directory
- The user must be a privileged user.
- The file must be writable by the user.

You can use the **mv** command to move files within the same file system or between file systems. Whether you are working in one file system or across file systems, the **mv** command copies the file to the target and deletes the original file. The **mv** command preserves in the new file the time of the most recent data modification, the time of the most recent access, the user ID, the group ID, and the file mode of the original file. For symbolic links, the **mv** command preserves only the owner and group of the link itself.

If it is unable to preserve the owner and group ID, the **mv** command clears S_ISUID and S_ISGID bits in the target. The **mv** command prints a diagnostic message to stderr if it is unable to clear these bits, though the exit code is not affected.

The **mv** command modifies either the source file or the destination path if the command is prematurely terminated.

Note: The **mv** command supports the **—** (dash, dash) parameter as a delimiter that indicates the end of the flags.

Flags

Attention: The **mv** command can overwrite many existing files unless you specify the **-i** flag. The **-i** flag prompts you to confirm before it overwrites a file. If both the **-f** and **-i** flags are specified in combination, the last flag specified takes precedence.

-E The **-E** option requires one of the following arguments. If you omit the **-E** option, **warn** is the default behavior.

force Fails the **mv** operation on a file if the fixed extent size or space reservation of the file cannot be preserved.

ignore Ignores any errors in preserving extent attributes.

warn Issues a warning if the space reservation or the fixed extent size of the file cannot be preserved.

-f Does not prompt you before overwriting an existing file.

-i Prompts you before moving a file or directory to an existing path name by displaying the name of the file followed by a question mark. If you answer with a line starting with `y` or the locale's equivalent of a `y`, the move continues. Any other reply prevents the move from occurring.

Examples

1. To rename a file, enter:

   ```
   mv appendix apndx.a
   ```

   This command renames `appendix` to `apndx.a`. If a file named `apndx.a` already exists, its old contents are replaced with those of `appendix`.

2. To move a directory, enter:

   ```
   mv book manual
   ```

   This command moves all files and directories under `book` to the directory named `manual`, if `manual` exists. Otherwise, the directory `book` is renamed `manual`.

3. To move a file to another directory and give it a new name, enter:

   ```
   mv intro manual/chap1
   ```

   This command moves `intro` to `manual/chap1`. The name `intro` is removed from the current directory, and the same file appears as `chap1` in the directory `manual`.

4. To move a file to another directory, keeping the same name, enter:

   ```
   mv chap3 manual
   ```

   This command moves `chap3` to `manual/chap3`

   Note: Examples 1 and 3 name two files, example 2 names two existing directories, and example 4 names a file and a directory.

5. To move several files into another directory, enter:

   ```
   mv chap4 jim/chap5 /home/manual
   ```

   This command moves the `chap4` file to the `/home/manual/chap4` file directory and the `jim/chap5` file to the `/home/manual/chap5` file.

6. To use the **mv** command with pattern-matching characters, enter:

   ```
   mv manual/* .
   ```

This command moves all files in the `manual` directory into the current directory . (period), retaining the names they had in `manual`. This move also empties `manual`. You must type a space between the asterisk and the period.

> **Note:** Pattern-matching characters expand names of existing files only. For example, the command `mv intro man*/chap1` does not work if the file `manual/chap1` does not exist.

Exit Status

0 All input files were moved successfully.
>0 An error occurred.

Files

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

Related Information

The **chmod** command, **ln** command, **rm** command.

The **rename** subroutine.

Files Overview in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

Input and Output Redirection Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

---

mvdir Command

Purpose

Moves (renames) a directory.

Syntax

mvdir *Directory1 Directory2*

Description

The **mvdir** command renames directories within a file system. To use the **mvdir** command, you must have write permission to *Directory1* and *Directory2* as well as in the parent directories.

The *Directory1* parameter must name an existing directory. If *Directory2* does not exist, *Directory1* is moved to *Directory2*. If *Directory2* exists, *Directory1* becomes a subdirectory of *Directory2*. Neither directory can be a subset of the other.

The **mvdir** Command can also be used to move or rename files. If the *Directory1* parameter is an existing file name and the *Directory2* parameter is an existing directory name, the file specified by *Directory1* is moved to the directory specified by *Directory2*. If the *Directory1* parameter is an existing file name and the *Directory2* parameter does not yet exist, *Directory2* replaces the file name *Directory1*. If both are existing file names, the file specified by *Directory1* is renamed *Directory2*, and the existing *Directory2* is removed.

The **mv** command provides the same functionality as the **mvdir** command.

Example

To rename or move a directory to another location, enter:

```
mvdir appendixes manual
```

If `manual` does not exist, this renames the `appendixes` directory to `manual`.

If a directory named `manual` already exists, this moves `appendixes` and its contents to `manual/appendixes`. In other words, `appendixes` becomes a subdirectory of `manual`.

Files

/usr/sbin/mvdir	Contains the **mvdir** command.

Related Information

The **mkdir** command, **mv** command.

The **rename** subroutine.

Directory Overview in *AIX 5L Version 5.2 System User's Guide: Operating System and Devices*.

File and Directory Access Modes in the *AIX 5L Version 5.2 System User's Guide: Operating System and Devices* introduces file ownership and permissions to access files and directories.

mvfilt Command

Purpose

Moves a filter rule.

Syntax

mvfilt -v 4|6 -p *p_fid* **-n** *n_fid*

Description

Use the **mvfilt** command to change the position of a filter rule in the filter rule table.

Flags

-v	IP version of the filter rule. The value **4** specifies IP version 4 and the value **6** specifies IP version 6.
-p	Filter rule ID. It specifies the previous position of the rule in the filter rule table. For IP version 4, the value of **1** is invalid since the first filter rule is unmoveable.
-n	Filter rule ID. It specifies the new position of the rule in the filter rule table after the move. For IP version 4, the value of **1** is invalid since the first filter rule is reserved and thus is unmoveable.

mvt Command

Purpose

Typesets English-language view graphs and slides.

Syntax

mvt [**-a**] [**-c**] [**-e**] [**-g**] [**-p**] [**-t**] [**-z**] [**-T***Name* | **-D***Destination*] [*File* ... | **-**]

Description

The **mvt** command typesets its input with the **mv** macro package for view graphs and slides in a manner similar to the **mmt** command. The **mvt** command has flags to specify preprocessing by the **tbl**, **eqn**, **pic**, **cw**, and **grap** commands. The flags you select determine which pipelines, flags, and parameters are generated for the **troff** command and the macro package.

The **mvt** command, unlike the **troff** command, automatically pipes its output to a postprocessor, unless specifically requested not to do so. The user should not specify a postprocessor when using the **mvt** command. The path that the **mvt** command takes is as follows:

1. The **-z** flag (no postprocessor is used).
2. The **-T***Name* flag.
3. The **TYPESETTER** environment variable is read.
4. The default is set to **ibm3816**.

File specifies the file that the **mvt** command formats.

Flags

Flags can occur in any order, but they must be displayed before the *File* parameter. If no file is specified, the **mvt** command prints a list of its flags.

-a	Displays readable **troff** output to the terminal.
-c	Calls the **cw** command.
-e	Calls the **eqn** command; also causes the **eqn** command to read the **/usr/share/lib/pub/eqnchar** file.
-g	Calls the **grap** command, which in turn calls the **pic** command.
-p	Calls the **pic** command.
-t	Calls the **tbl** command.
-z	Calls no output filter (or postprocessor) to process or redirect the output of the **troff** command.
-D*Destination*	Directs the output to the specified device destination. Supported value for the *Destination* variable is **4014**, which is the Tektronix 4014 terminal by way of the **tc** command.
-T*Name*	Creates output for the **troff** device as specified by the *Name* variable. The output is sent through the appropriate postprocessor. The default is **ibm3816**.
-	Forces input to be read from standard input.

Any other parameters or flags that you give the **mvt** command (such as the **-a** flag) are passed to the **troff** command.

The **mvt** command reads standard input when you specify the **-** (minus) flag instead of the *File* parameter.

Use the **-o***List* flag of the **troff** command to specify ranges of pages to be output.

> **Note:** If you call the **mvt** command with one or more of the **-e**, **-c**, **-t**, **-p**, **-g**, or **-** flags, together with the **-o***List* flag of the **troff** command, you may receive a `broken pipe` message. This occurs if you do not specify the last page of the document in the *List* variable. This broken pipe message is not an indication of any problem and can be ignored.

Environment Variables

TYPESETTER Contains information about a particular printing device.

Files

/usr/share/lib/pub/eqnchar	Contains special character definitions.

Related Information

The **cw** command, **eqn** command, **grap** command, **mmt** command, **pic** command, **tbl** command, **tc** command, **troff** command.

The **eqnchar** file format.

The article ″mv Macro Package for the mvt and troff Commands″ in the **troff** Command.

---

mwm Command

Purpose

Runs the AIXwindows Window Manager (MWM).

Syntax

mwm -display *Host:Display:ScreenID* **-xrm** *ResourceString* **-multiscreen -name** *Name* **-screens** *Name* [*Name* ...]

Description

The **mwm** command runs the AIXwindows Window Manager (MWM) and is often started by a display or session manager. The AIXwindows Window Manager (MWM) is an X Window System client that provides window management functionality and some session management functionality. It provides functions that facilitate control (by the user and the programmer) of elements of window states such as placement, size, icon or normal display, and input-focus ownership. It also provides session management functions such as stopping a client.

The appearance and behavior of the window manager can be altered by changing the configuration of specific resources. Resources are defined under X Defaults .

By default, the **mwm** command manages only the single screen specified by the **-display** option or the **DISPLAY** environment variable (by default, screen 0). If the **-multiscreen** option is specified or if the **multiScreen** resource is True, the **mwm** command tries to manage all the screens on the display.

When the **mwm** command is managing multiple screens, the **-screens** option can be used to give each screen a unique resource name. The names are separated by blanks, for example, **-screens mwm0 mwm1**. If there are more screens than names, resources for the remaining screens are retrieved using the first name. By default, the screen number is used for the screen name.

For information on windows, icons, resources, events, button and key bindings, menus, and variables, see the following sections:

- Windows
- Icons
- Icon Box
- Component Appearance Resources
- General Appearance and Behavior Resources
- Client-Specific Resources

- Window Manager Event Specification
- Button Bindings
- Key Bindings
- Menu Panes
- Environment
- Related Information

Flags

-display *Host*:*Display*:*ScreenID* Specifies the display to use. The **-display** option has the following parameters:

Host Specifies the host name of a valid system on the network. Depending on the situation, this could be the host name of the user or the host name of a remote system.

Display Specifies the number (usually 0) of the display on the system on which the output is to be displayed.

ScreenID
Specifies the number of the screen where the output is to be displayed. This number is 0 for single-screen systems.

-xrm *ResourceString* Enables the named resources when starting the **mwm** command.

-multiscreen Causes the **mwm** command to manage all screens on the display. The default is to manage only a single screen.

-name *Name* Causes the **mwm** command to retrieve its resources using the specified name, as in *Name*Resource*.

-screens *Name* [*Name* [...]] Specifies the resource names to use for the screens managed by MWM. If MWM is managing a single screen, only the first name in the list is used. If multiple screens are being managed, the names are assigned to the screens in order, starting with screen 0. For example, screen 0 gets the first name and screen 1 gets the second name.

Windows

Default window manager window frames have the following distinct components with associated functions:

title area In addition to displaying the client's title, the title area is used to move the window. To move the window, place the pointer over the title area, press button 1 and drag the window to a new location. A wire frame is moved during the drag to indicate the new location. When the button is released, the window is moved to the new location.

title bar The title bar includes the title area, the **Minimize** button, the **Maximize** button, and the **Window Menu** button. In shaped windows, such as round windows, the title bar floats above the window.

Minimize button To turn the window into an icon, click button 1 on the **Minimize** button (the frame box with a small square in it).

Maximize button To make the window fill the screen (or enlarge to the largest size allowed by the configuration files), click button 1 on the **Maximize** button (the frame box with a large square in it).

Window Menu button

The **Window Menu** button is the frame box with a horizontal bar in it. To pull down the window menu, press button 1. While pressing the button, drag the pointer on the menu to your selection and release the button when your selection is highlighted. Pressing button 3 in the title bar or resize border handles also posts the window menu. Alternately, you can click button 1 to pull down the menu and keep it posted; then position the pointer and select. You can also post the window menu by pressing the Shift+Esc or Alt+Space key sequence. Double-clicking button 1 with the pointer on the **Window Menu** button closes the window. The following table lists the contents of the window menu:

Default Window Menu

Selection	Accelerator	Description
Restore	Alt+F5	Restores the window to its size before minimizing or maximizing.
Move	Alt+F7	Allows the window to be moved with keys or mouse.
Size	Alt+F8	Allows the window to be resized.
Minimize	Alt+F9	Turns the window into an icon.
Maximize	Alt+F10	Makes the window fill the screen.
Lower	Alt+F3	Moves window to bottom of window stack.
Close	Alt+F4	Causes client to stop.

resize border handles

To change the size of a window, move the pointer over a resize border handle (the cursor changes), press button 1, and drag the window to a new size. When the button is released, the window is resized. While dragging is being done, a rubber-band outline is displayed to indicate the new window size.

matte

An optional matte decoration can be added between the client area and the window frame. A matte is not actually part of the window frame. There is no functionality associated with a matte.

Icons

Icons are small graphic representations of windows. A window can be iconified (minimized) using the **Minimize** button on the window frame. Icons provide a way to reduce clutter on the screen.

Pressing the left mouse button when the pointer is over an icon causes the icon's window menu to open. Releasing the button (press + release without moving mouse = click) causes the menu to stay posted. The menu contains the following selections:

Icon Window Menu

Selection	Accelerator	Description
Restore	Alt+F5	Opens the associated window.
Move	Alt+F7	Allows the icon to be moved with keys.
Size	Alt+F8	Inactive (not an option for icons).
Minimize	Alt+F9	Inactive (not an option for icons).
Maximize	Alt+F10	Opens the associated window and makes it fill the screen.
Lower	Alt+F3	Moves icon to bottom of icon stack.
Close	Alt+F4	Removes client from window manager management.

Pressing button 3 over an icon also causes the icon's window menu to open. To make a menu selection, drag the pointer over the menu and release button 3 when the desired item is highlighted.

Double-clicking button 1 on an icon calls the **f.restore_and_raise** function and restores the icon's associated window to its previous state. For example, if a maximized window is iconified, double-clicking button 1 restores it to its maximized state. Double-clicking button 1 on the icon box's icon opens the icon

box and allow access to the contained icons. (Double-clicking a mouse button is a quick way to perform a function.) Pressing the Shift+Esc key sequence or the pop-up Menu key causes the icon window menu of the currently selected icon to open.

Icon Box

When icons begin to clutter the screen, they can be packed into an icon box. (To use an icon box, the window manager must be started with the icon box configuration already set.) The icon box is a window manager window that holds client icons. It includes one or more scroll bars when there are more window icons than the icon box can show at the same time.

Icons in the icon box can be manipulated with the mouse. The following button action descriptions summarize the behavior of this interface. Button actions apply whenever the pointer is on any part of the icon. Double-clicking an icon in the icon box calls the **f.restore_and_raise** function.

Button Action	Description
Button 1 click	Selects the icon.
Button 1 double-click	Normalizes (opens) the associated window.
Button 1 double-click	Raises an already *open* window to the top of the stack.
Button 1 drag	Moves the icon.
Button 3 press	Causes the menu for that icon to open.
Button 3 drag	Highlights items as the pointer moves across the menu.

Pressing mouse button 3 when the pointer is over an icon causes the menu for that icon to open.

Icon Menu for Icon Box

Selection	Accelerator	Description
Restore	Alt+F5	Opens the associated window (if not already open).
Move	Alt+F7	Allows the icon to be moved with keys.
Size	Alt+F8	Inactive.
Minimize	Alt+F9	Inactive.
Maximize	Alt+F10	Opens the associated window (if not already open) and maximizes its size.
Lower	Alt+F3	Inactive.
Close	Alt+F4	Removes client from window manager management.

To pull down the window menu for the icon box itself, press button 1 with the pointer over the menu button for the icon box. The window menu of the icon box differs from the window menu of a client window: The **Close** selection is replaced with the **PackIcons** (Shift+Alt+F7) selection. When selected, the **PackIcons** option packs the icons in the box to achieve neat rows with no empty slots.

You can also post the window menu by pressing the Shift+Esc or Alt+Space key sequence. Pressing the pop-up Menu key causes the icon window menu of the currently selected icon to open.

Input Focus

The **mwm** command supports (by default) a keyboard input focus policy of *explicit selection*. This means when a window is selected to get keyboard input, it continues to get keyboard input until the window is withdrawn from window management, another window is explicitly selected to get keyboard input, or the window is iconified. Several resources control the input focus. The client window with the keyboard input focus has the active window appearance with a visually distinct window frame.

The following table and key action descriptions summarize the keyboard input focus selection behavior:

Button Action	Object	Function Description
Button 1 press	Window or window frame	Keyboard focus selection
Button 1 press	Icon	Keyboard focus selection

Key Action	Function Description
Alt+Tab	Moves the input focus to next window in the window stack.
Alt+Shift+Tab	Moves the input focus to the previous window in the window stack (available only in explicit focus mode).

Window Stacking

There are two types of window stacks: global window stacks and an application's local family window stack.

The global stacking order of windows can be changed as a result of setting the keyboard input focus, iconifying a window, or performing a window manager window stacking function. When keyboard focus policy is explicit the default value of the **focusAutoRaise** resource is True. This causes a window to be raised to the top of the stack when it receives input focus, for example, by pressing button 1 on the title bar. The key actions defined in the preceding list raises the window receiving focus to the top of the stack.

In pointer mode, the default value of the **focusAutoRaise** resource is False; that is, the window stacking order is not changed when a window receives keyboard input focus. The following key actions can be used to cycle through the global window stack:

Key Action	Function Description
Alt+Esc	Places top window on bottom of stack.
Alt+Shift+Esc	Places bottom window on top of stack.

By default, a window's icon is placed on the bottom of the stack when the window is iconified; however, the default can be changed by the **lowerOnIconify** resource.

Transient windows (secondary windows such as dialog boxes) stay above their parent windows by default. However, an application's local family stacking order can changed to allow a transient window to be placed below its parent top-level window. The following parameter values show the modification of the stacking order for the **f.lower** function:

f.lower	Lowers the transient window within the family (staying above the parent) and lowers the family in the global window stack.
f.lower [**within**]	Lowers the transient window within the family (staying above the parent) but does not lower the family in the global window stack.
f.lower [**freeFamily**]	Lowers the window separate from its family stack (below the parent), but does not lower the family in the global window stack.

The **within** and **freeFamily** parameter values can also be used with the **f.raise** and **f.raise_lower** functions.

X Defaults

The **mwm** command is configured from its resource database. This database is built from the following sources. They are listed in order of precedence.

1. **mwm** command line options
2. **XENVIRONMENT** variable or **$HOME/.Xdefaults-host**

3. **RESOURCE_MANAGER** root window property or **$HOME/.Xdefaults**
4. **$HOME/Mwm**
5. **/usr/lib/X11/app-defaults/Mwm**.

The **/usr/lib/X11/app-defaults/Mwm** and **$HOME/Mwm** file names represent customary locations for these files. The actual location of the systemwide class resource file might depend on the **XFILESEARCHPATH** environment variable and the current language environment. The actual location of the user-specific class resource file might depend on the **XUSERFILESEARCHPATH** and **XAPPLRESDIR** environment variables and the current language environment.

Entries in the resource database can refer to other resource files for specific types of resources. These include files that contain bitmaps, fonts, and **mwm**-specific resources such as menus and behavior specifications (for example, button and key bindings).

Mwm is the resource class name of the **mwm** command and **mwm** is the resource name used by the **mwm** command to look up resources. (For looking up resources of multiple screens, the **-screens** command-line option specifies resource names such as **mwm_b+w** and **mwm_color**.) In the following discussion of resource specification, ″Mwm″ and ″mwm″ (and the aliased **mwm** resource names) can be used interchangeably, but ″mwm″ takes precedence over ″Mwm″. The **mwm** command uses the following types of resources:

component appearance resource set	These resources specify appearance attributes of window manager user-interface components. They can be applied to the appearance of window manager menus, feedback windows (for example, the window reconfiguration feedback window), client window frames, and icons.
frame and icon component resource set	This subset of component appearance resources specifies attributes that are specific to frame and icon components.
general appearance and behavior resource set	These resources specify the **mwm** command appearance and behavior (for example, window management policies). They are not set separately for different **mwm** command user-interface components.
client-specific resource set	These **mwm** resources can be set for a particular client window or class of client windows. They specify client-specific icon and client window frame appearance and behavior.

Resource identifiers can be either a resource name (for example, **foreground**) or a resource class (for example, **Foreground**). If the value of a resource is a file name and if the file name is prefixed by the ~/ (tilde followed by a slash) characters, it is relative to the path contained in the **HOME** environment variable (generally the user's home directory).

Component Appearance Resources

The syntax for specifying component appearance resources that apply to window manager icons, menus, and client window frames is as follows:

Mwm*ResourceID*

For example, **Mwm*foreground** is used to specify the foreground color for the **mwm** command menus, icons, client window frames, and feedback dialogs.

The syntax for specifying component appearance resources that apply to a particular **mwm** component is as follows:

Mwm*[*Menu*|*Icon*|*Client*|*Feedback*]**ResourceID*

If *Menu* is specified, the resource is applied only to Mwm menus; if *Icon* is specified, the resource is applied to icons; and if *Client* is specified, the resource is applied to client window frames. For example, **Mwm*** *Icon* ***foreground** is used to specify the foreground color for the **mwm** command icons, **Mwm*** *Menu* ***foreground** specifies the foreground color for the **mwm** command menus, and **Mwm*** *Client* ***foreground** is used to specify the foreground color for the **mwm** command client window frames.

The appearance of the title area of a client window frame (including window management buttons) can be separately configured. The syntax for configuring the title area of a client window frame is as follows:

Mwm* *Client* * *Title* * *ResourceID*

For example, **Mwm*** *Client* * *Title* ***foreground** specifies the foreground color for the title area. Defaults for title area resources are based on the values of the corresponding client window frame resources.

The appearance of menus can be configured based on the name of the menu. The syntax for specifying menu appearance by name is as follows:

Mwm* *Menu* * *MenuName* * *ResourceID*

For example, **Mwm*** *Menu* *`MyMenu` ***foreground** specifies the foreground color for the menu named `MyMenu`.

The user can also specify resources for window manager menu components (the gadgets that comprise the menu). These may include, for example, a menu title, a title separator, one or more buttons, and separators. If a menu contains more than one instance of a class, such as multiple **PushButtonGadget** gadgets, the name of the first instance is **PushButtonGadget1**, the second is **PushButtonGadget2**, and so on. The following list identifies the naming conventions used for window manager menu components:

TitleName	Menu title LabelGadget
TitleSeparator	Menu title SeparatorGadget
CascadeButtonGadget*n*	CascadeButtonGadget
PushButtonGadget*n*	PushButtonGadget
SeparatorGadget*n*	SeparatorGadget

The following component appearance resources that apply to all window manager parts can be specified.

Component Appearance Resource Set

Name	Class	Value Type	Default
background	**Background**	color	varies[1]
backgroundPixmap	**BackgroundPixmap**	string[2]	varies[1]
bottomShadowColor	**Foreground**	color	varies[1]
bottomShadowPixmap	**BottomShadowPixmap**	string[2]	varies[1]
fontList	**FontList**	string[3]	″fixed″
foreground	**Foreground**	color	varies[1]
saveUnder	**SaveUnder**	True or False	False
topShadowColor	**Background**	color	varies[1]
topShadowPixmap	**TopShadowPixmap**	string[2]	varies[1]
background	**Background**	color	varies[1]
backgroundPixmap	**BackgroundPixmap**	string[2]	varies[1]
bottomShadowColor	**Foreground**	color	varies[1]

Name	Class	Value Type	Default
bottomShadowPixmap	**BottomShadowPixmap**	string[2]	varies[1]
fontList	**FontList**	string[3]	″fixed″
foreground	**Foreground**	color	varies[1]
saveUnder	**SaveUnder**	True or False	False
topShadowColor	**Background**	color	varies[1]
topShadowPixmap	**TopShadowPixmap**	string[2]	varies[1]

Notes:

1. The default is chosen based on the visual type of the screen.
2. Image name.
3. X Version 11 Release 4 (X11R4) font description.

background (class **Background**)
Specifies the background color. Any legal X color can be specified. The default value is chosen based on the visual type of the screen.

backgroundPixmap (class **BackgroundPixmap**)
Specifies the background pixmap of the **mwm** decoration when the window is inactive (does not have the keyboard focus). The default value is chosen based on the visual type of the screen.

bottomShadowColor (class **Foreground**)
Specifies the bottom shadow color. This color is used for the lower and right bevels of the window manager decoration. Any legal X color can be specified. The default value is chosen based on the visual type of the screen.

bottomShadowPixmap (class **BottomShadowPixmap**)
Specifies the bottom shadow pixmap. This pixmap is used for the lower and right bevels of the window manager decoration. The default is chosen based on the visual type of the screen.

fontList (class **FontList**)
Specifies the font used in the window manager decoration. The character encoding of the font needs to match the character encoding of the strings that are used. The default is the fixed value.

foreground (class **Foreground**)
Specifies the foreground color. The default is chosen based on the visual type of the screen.

saveUnder (class **SaveUnder**)
Controls the repainting of windows that are uncovered after being obscured. This resource indicates whether *save unders* are used for **mwm** components. For this to have any effect, save unders must be implemented by the X server. If save unders are implemented, the X server saves the contents of windows obscured by windows that have the save under attribute set. If the **saveUnder** resource is True, the **mwm** command sets the save under attribute on the window manager frame of any client that has it set. If the **saveUnder** resource is False, save unders is not used on any window manager frames. The default value is False.

topShadowColor (class **Background**)
Specifies the top shadow color. This color is used for the upper and left bevels of the window manager decoration. The default is chosen based on the visual type of the screen.

topShadowPixmap (class **TopShadowPixmap**)
Specifies the top shadow pixmap. This pixmap is used for the upper and left bevels of the window manager decoration. The default is chosen based on the visual type of the screen.

Frame and Icon Component Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

Name	Class	Value Type	Default
activeBackground	**Background**	color	varies[1]
activeBackground-Pixmap	**BackgroundPixmap**	string[2]	varies[1]
activeBottomShadow-Color	**Foreground**	color	varies[1]
activeBottomShadow-Pixmap	**BottomShadow-Pixmap**	string[2]	varies[1]
activeForeground	**Foreground**	color	varies[1]
activeTopShadowColor	**Background**	color	varies[1]
activeTopShadowPixmap	**TopShadowPixmap**	string[2]	varies[1]
activeBackground	**Background**	color	varies[1]
activeBackgroundPixmap	**BackgroundPixmap**	string[2]	varies[1]
activeBottomShadowColor	**Foreground**	color	varies[1]
activeBottomShadowPixmap	**BottomShadowPixmap**	string[2]	varies[1]
activeForeground	**Foreground**	color	varies[1]
activeTopShadowColor	**Background**	color	varies[1]
activeTopShadowPixmap	**TopShadowPixmap**	string[2]	varies[1]

Notes:

1. The default is chosen based on the visual type of the screen.
2. Image name.

activeBackground (class **Background**)
Specifies the background color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeBackgroundPixmap (class **BackgroundPixmap**)
Specifies the background pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeBottomShadowColor (class **Foreground**)
Specifies the bottom shadow color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeBottomShadowPixmap (class **BottomShadowPixmap**)
Specifies the bottom shadow pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeForeground (class **Foreground**)
Specifies the foreground color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeTopShadowColor (class **Background**)
Specifies the top shadow color of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

activeTopShadowPixmap (class **TopShadowPixmap**)
Specifies the top shadow pixmap of the **mwm** decoration when the window is active (has the keyboard focus). The default is chosen based on the visual type of the screen.

General Appearance and Behavior Resources

The syntax for specifying general appearance and behavior resources is as follows:

Mwm*ResourceID*

For example, **Mwm*keyboardFocusPolicy** specifies the window manager policy for setting the keyboard focus to a particular client window.

General Appearance and Behavior Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

Name	Class	Value Type	Default
autoKeyFocus	**AutoKeyFocus**	True or False	True
autoRaiseDelay	**AutoRaiseDelay**	millisec.	500
bitmapDirectory	**BitmapDirectory**	directory	**/usr/include/X11/**bitmaps
buttonBindings	**ButtonBindings**	string	″DefaultButton-Bindings″
cleanText	**CleanText**	True or False	True
clientAutoPlace	**ClientAutoPlace**	True or False	True
colormapFocusPolicy	**ColormapFocus-Policy**	string	keyboard
configFile	**ConfigFile**	file	**.mwmrc**
deiconifyKeyFocus	**DeiconifyKeyFocus**	True or False	True
doubleClickTime	**DoubleClickTime**	millisec.	multiclick time
enableWarp	**enableWarp**	True or False	True
enforceKeyFocus	**EnforceKeyFocus**	True or False	True
fadeNormalIcon	**FadeNormalIcon**	True or False	False
feedbackGeometry	**FeedbackGeometry**	string	center on screen
frameBorderWidth	**FrameBorderWidth**	pixels	varies
iconAutoPlace	**IconAutoPlace**	True or False	True
iconBoxGeometry	**IconBoxGeometry**	string	6x1+0-0
iconBoxName	**IconBoxName**	string	iconbox
iconBoxSBDisplayPolicy	**IconBoxSBDisplayPolicy**	string	all
iconBoxTitle	**IconBoxTitle**	XmString	Icons
iconClick	**IconClick**	True or False	True
iconDecoration	**IconDecoration**	string	varies
iconImage-Maximum	**IconImage-Maximum**	wxh	50x50
iconImageMinimum	**IconImageMinimum**	wxh	16x16
iconPlacement	**IconPlacement**	string	left bottom

Name	Class	Value Type	Default
iconPlacementMargin	**IconPlacementMargin**	pixels	varies
interactivePlacement	**InteractivePlacement**	True or False	False
keyBindings	**KeyBindings**	string	″DefaultKeyBindings″
keyboardFocusPolicy	**KeyboardFocusPolicy**	string	explicit
limitResize	**LimitResize**	True or False	True
lowerOnIconify	**LowerOnIconify**	True or False	True
maximumMaximuSize	**MaximumMaximuSize**	wxh (pixels)	2X screen w&h
moveOpaque	**MoveOpaque**	True or False	False
moveThreshold	**MoveThreshold**	pixels	4
multiScreen	**MultiScreen**	True or False	False
passButtons	**PassButtons**	True or False	False
PassSelectButton	**PassSelectButton**	True or False	True
positionIsFrame	**PositionIsFrame**	True or False	True
positionOnScreen	**PositionOnScreen**	True or False	True
quitTimeout	**QuitTimeout**	millisec.	1000
raiseKeyFocus	**RaiseKeyFocus**	True or False	False
resizeBorderWidth	**ResizeBorder-Width**	pixels	varies
resizeCursors	**ResizeCursors**	True or False	True
screens	**Screens**	string	varies
showFeedback	**ShowFeedback**	string	all
startupKeyFocus	**StartupKeyFocus**	True or False	True
transientDecoration	**Transient-Decoration**	string	menu title
transientFunctions	**Transient-Functions**	string	-minimize -maximize
useIconBox	**UseIconBox**	True or False	False
wMenuButtonClick	**WMenuButtonClick**	True or False	True
wMenuButtonClick2	**WMenuButtonClick2**	True or False	True
autoKeyFocus	**AutoKeyFocus**	True or False	True
autoRaiseDelay	**AutoRaiseDelay**	millisec.	500
bitmapDirectory	**BitmapDirectory**	directory	**/usr/include/X11/**bitmaps
buttonBindings	**ButtonBindings**	string	″DefaultButtonBindings″

Name	Class	Value Type	Default
cleanText	**CleanText**	True or False	True
clientAutoPlace	**ClientAutoPlace**	True or False	True
colormapFocusPolicy	**ColormapFocusPolicy**	string	keyboard
configFile	**ConfigFile**	file	**.mwmrc**
deiconifyKeyFocus	**DeiconifyKeyFocus**	True or False	True
doubleClickTime	**DoubleClickTime**	millisec.	multiclick time
enableWarp	**enableWarp**	True or False	True
enforceKeyFocus	**EnforceKeyFocus**	True or False	True
fadeNormalIcon	**FadeNormalIcon**	True or False	False
feedbackGeometry	**FeedbackGeometry**	string	center on screen
frameBorderWidth	**FrameBorderWidth**	pixels	varies
iconAutoPlace	**IconAutoPlace**	True or False	True
iconBoxGeometry	**IconBoxGeometry**	string	6x1+0-0
iconBoxName	**IconBoxName**	string	iconbox
iconBoxSBDisplayPolicy	**IconBoxSBDisplayPolicy**	string	all
iconBoxTitle	**IconBoxTitle**	XmString	Icons
iconClick	**IconClick**	True or False	True
iconDecoration	**IconDecoration**	string	varies
iconImageMaximum	**IconImageMaximum**	wxh	50x50
iconImageMinimum	**IconImageMinimum**	wxh	16x16
iconPlacement	**IconPlacement**	string	left bottom
iconPlacementMargin	**IconPlacementMargin**	pixels	varies
interactivePlacement	**InteractivePlacement**	True or False	False
keyBindings	**KeyBindings**	string	″DefaultKeyBindings″
keyboardFocusPolicy	**KeyboardFocusPolicy**	string	explicit
limitResize	**LimitResize**	True or False	True
lowerOnIconify	**LowerOnIconify**	True or False	True
maximumMaximumSize	**MaximumMaximumSize**	wxh (pixels)	2X screen w&h
moveOpaque	**MoveOpaque**	True or False	False
moveThreshold	**MoveThreshold**	pixels	4
multiScreen	**MultiScreen**	True or False	False

Name	Class	Value Type	Default
passButtons	**PassButtons**	True or False	False
PassSelectButton	**PassSelectButton**	True or False	True
positionIsFrame	**PositionIsFrame**	True or False	True
positionOnScreen	**PositionOnScreen**	True or False	True
quitTimeout	**QuitTimeout**	millisec.	1000
raiseKeyFocus	**RaiseKeyFocus**	True or False	False
resizeBorderWidth	**ResizeBorderWidth**	pixels	varies
resizeCursors	**ResizeCursors**	True or False	True
screens	**Screens**	string	varies
showFeedback	**ShowFeedback**	string	all
startupKeyFocus	**StartupKeyFocus**	True or False	True
transientDecoration	**TransientDecoration**	string	menu title
transientFunctions	**TransientFunctions**	string	-minimize -maximize
useIconBox	**UseIconBox**	True or False	False
wMenuButtonClick	**WMenuButtonClick**	True or False	True
wMenuButtonClick2	**WMenuButtonClick2**	True or False	True

Note: Abbreviations for value types:
`w = width`
`h = height`
`millisec = milliseconds`			

autoKeyFocus (class **AutoKeyFocus**)

Controls whether the focus is set to the previous window that had the focus. If the **autoKeyFocus** resource is given a value of True and a window with the keyboard input focus is withdrawn from window management or is iconified, the focus is set to the previous window that had the focus. If the value given is False, there is no automatic setting of the keyboard input focus. It is recommended that both the **autoKeyFocus** resource and the **startupKeyFocus** resource be set to a value of True to work with tear-off menus. The default value is True. This resource is available only when the keyboard input focus policy is set to the explicit value.

autoRaiseDelay (class **AutoRaiseDelay**)

Specifies the amount of time in milliseconds (ms) that the **mwm** command waits before raising a window after it gets the keyboard focus. The default value of this resource is 500 (milliseconds). This resource is available only when the **focusAutoRaise** resource is True and the keyboard focus policy is the pointer value.

bitmapDirectory (class **BitmapDirectory**)
Identifies a directory to be searched for bitmaps referenced by **mwm** resources. This directory is searched if a bitmap is specified without an absolute path name. The default value for this resource is **/usr/include/X11/bitmaps**. The **/usr/include/X11/bitmaps** directory represents the customary locations for this directory. The actual location of this directory may vary on some systems. If the bitmap is not found in the specified directory, the **XBMLANGPATH** environment variable is searched.

buttonBindings (class **ButtonBindings**)
Identifies the set of button bindings for window management functions. The named set of button bindings is specified in the **mwm** resource description file. These button bindings are merged with the built-in default bindings. The default value for this resource is **DefaultButtonBindings**.

cleanText (class **CleanText**)
Controls the display of window manager text in the client title and feedback windows. If the default value of True is used, the text is drawn with a clear (no stipple) background. This makes text easier to read on monochrome systems where a **backgroundPixmap** is specified. Only the stippling in the area immediately around the text is cleared. If False, the text is drawn directly on top of the existing background.

clientAutoPlace (class **ClientAutoPlace**)
Determines the position of a window when the window does not have a user-specified position. With a value of True, windows are positioned with the top left corners of the frames offset horizontally and vertically. A value of False causes the currently configured position of the window to be used. In either case, the **mwm** command attempts to place the windows totally on-screen. The default value is True.

colormapFocusPolicy (class **ColormapFocusPolicy**)
Indicates the colormap focus policy that is to be used. If the resource value is explicit, a colormap selection action is done on a client window to set the colormap focus to that window. If the value is pointer, the client window containing the pointer has the colormap focus. If the value is keyboard, the client window that has the keyboard input focus has the colormap focus. The default value for this resource is keyboard.

configFile (class **ConfigFile**)

Contains the path name for an **mwm** resource description file.

If the path name begins with the ~/ characters, the **mwm** command considers it to be relative to the user's home directory (as specified by the **HOME** environment variable). If the **LANG** environment variable is set, the **mwm** command looks for **$HOME/$LANG/configFile**. If that file does not exist or if **LANG** is not set, **mwm** looks for **$HOME/configFile**.

If the **configFile** path name does not begin with the ~/ characters, **mwm** considers it to be relative to the current working directory.

If the **configFile** resource is not specified or if that file does not exist, the **mwm** command uses several default paths to find a configuration file. If the **LANG** environment variable is set, the **mwm** command looks for the configuration file first in the **$HOME/$LANG/.mwmrc** file. If that file does not exist or if the **LANG** environment variable is not set, the **mwm** command looks for the **$HOME/.mwmrc** file. If the **$HOME/.mwmrc** file does not exist and if the **LANG** environment variable is set, the **mwm** command next looks for the **/usr/lib/X11/$LANG/system.mwmrc** file. If the **/usr/lib/X11/$LANG/system.mwmrc** file does not exist or if the **LANG** environment variable is not set, the **mwm** command looks for **/usr/lib/X11/system.mwmrc**.

deiconifyKeyFocus (class **DeiconifyKeyFocus**)

Determines whether a window receives the keyboard input focus when it is deiconified (normalized). The default value is True. This resource applies only when the keyboard input focus policy is set to the explicit value.

doubleClickTime (class **DoubleClickTime**)

Sets the maximum time (in ms) between the clicks (button presses) that make up a double-click. The default value of this resource is the display's multiclick time.

enableWarp (class **EnableWarp**)

Causes the **mwm** command to *warp* the pointer to the center of the selected window during keyboard-controlled resize and move operations. Setting the value to False causes the **mwm** command to leave the pointer at its original place on the screen unless the user explicitly moves it with the cursor keys or pointing device. The default value of this resource is True.

enforceKeyFocus (class **EnforceKeyFocus**)

Determines whether the keyboard input focus is always explicitly set to selected windows even if there is an indication that they are *globally active* input windows. (An example of a globally active window is a scroll bar that can be operated without setting the focus to that client.) If the resource is False, the keyboard input focus is not explicitly set to globally active windows. The default value is True.

fadeNormalIcon (class **FadeNormalIcon**)

Determines whether an icon is unavailable whenever it is normalized (its window is opened). The default value is False.

feedbackGeometry (class **FeedbackGeometry**)

Sets the position of the move and resize feedback window. If this resource is not specified, the default is to place the feedback window at the center of the screen. The value of the resource is a standard window geometry string with the following syntax:

[=][{+-}*XOffset*{+-}*YOffset*]

frameBorderWidth (class **FrameBorderWidth**)
Specifies the width in pixels of a client window frame border without resize handles. The border width includes the three-dimensional (3-D) shadows. The default value is based on the size and resolution of the screen.

iconAutoPlace (class **IconAutoPlace**)
Indicates whether the window manager arranges icons in a particular area of the screen or places each icon where the window was when it was iconified. The True value indicates that icons are arranged in a particular area of the screen determined by the **iconPlacement** resource. The False value indicates that an icon is placed at the location of the window when it is iconified. The default is True.

iconBoxGeometry (class **IconBoxGeometry**)
Indicates the initial position and size of the icon box. The value of the resource is a standard window geometry string with the following syntax:

[=][*Width*x*Height*][{+-}*XOffset*{+-}*YOffset*]

If the offsets are not provided, the **iconPlacement** policy is used to determine the initial placement. The units for width and height are columns and rows.

The actual screen size of the icon box window depends on the **iconImageMaximum** (size) and **iconDecoration** resources. The default value for size is (6 times **iconWidth** + padding) wide by (1 times **iconHeight** + padding) high. The default value of the location is +0 -0.

iconBoxName (class **IconBoxName**)
Specifies the name that is used to look up icon box resources. The default name is **iconbox**.

iconBoxSBDisplayPolicy (class **IconBoxSBDisplayPolicy**)
Specifies the scroll bar display policy of the window manager in the icon box. The resource has the following three possible values: all, vertical, and horizontal. The default value, all, causes both vertical and horizontal scroll bars always to be displayed. The vertical value causes a single vertical scroll bar to be displayed in the icon box and sets the orientation of the icon box to horizontal (regardless of the **iconBoxGeometry** specification). The horizontal value causes a single horizontal scroll bar to be displayed in the icon box and sets the orientation of the icon box to vertical (regardless of the **iconBoxGeometry** specification).

iconBoxTitle (class **IconBoxTitle**)
Specifies the name that is used in the title area of the icon box frame. The default value is Icons.

iconClick (class **IconClick**)
Specifies whether the system menu is posted and remains posted when an icon is clicked. The default value is True.

iconDecoration (class **IconDecoration**)
Specifies the general icon decoration. The resource value is label (only the label part is displayed) or image (only the image part is displayed) or label image (both the label and image parts are displayed). A value of activelabel can also be specified to get a label (not truncated to the width of the icon) when the icon is selected. The default icon decoration for icon box icons is that each icon has a label part and an image part (label image). The default icon decoration for standalone icons is that each icon has an active label part, a label part, and an image part (activelabel, label, and image).

iconImageMaximum (class **IconImageMaximum**)
Specifies the maximum size of the icon image. The resource value is *Width* x *Height* (for example, 64x64). The maximum supported size is 128x128. The default value of this resource is 50x50.

iconImageMinimum (class **IconImageMinimum**)

Specifies the minimum size of the icon image. The resource value is *Width* x *Height* (for example, 32x50). The minimum supported size is 16x16. The default value of this resource is 16x16.

iconPlacement (class **IconPlacement**)

Specifies the icon placement scheme to be used. The resource value has the following syntax:

PrimaryLayout SecondaryLayout [Tight]

The layout values are described as one of the following:

top Lays out the icons from top to bottom.

bottom Lays out the icons from bottom to top.

left Lays out the icons from left to right.

right Lays out the icons from right to left.

A horizontal (vertical) layout value must not be used for both the *PrimaryLayout* and the *SecondaryLayout* (for example, do not use top for the *PrimaryLayout* and bottom for the *SecondaryLayout*). The *PrimaryLayout* indicates at the time an icon placement is done whether the icon is placed in a row or a column and the direction of placement. The *SecondaryLayout* indicates where to place new rows or columns.

For example, the top right value indicates that icons should be placed top to bottom on the screen and that columns should be added from right to left on the screen. The default placement is the left bottom value (icons are placed from left to right on the screen, with the first row on the bottom of the screen, and new rows added from the bottom of the screen to the top of the screen). A tight value places icons with zero spacing between icons. This value is useful for aesthetic reasons, as well as for terminals with small screens.

Following is a list of options for **iconPlacement** values:

Icon Placement
Appropriate Scheme

From left to right across the top of the screen, new rows below
Left top

From right to left across the top of the screen, new rows below
Right top

From left to right across the bottom of the screen, new rows above
Left bottom

From right to left across the bottom of the screen, new rows above
Right bottom

From bottom to top along the left of the screen, new columns to right
Bottom left

From bottom to top along the right of the screen, new columns to left
Bottom right

From top to bottom along the left of the screen, new columns to right
Top left

From top to bottom along the right of the screen, new columns to left
Top right

iconPlacementMargin (class **IconPlacementMargin**)
Sets the distance between the edge of the screen and the icons that are placed along the edge of the screen. The value should be greater than or equal to 0. A default value is used if the value specified is invalid. The default value for this resource is equal to the space between icons as they are placed on the screen (this space is based on maximizing the number of icons in each row and column).

interactivePlacement (class **InteractivePlacement**)
Controls the initial placement of new windows on the screen. If the value is True, the pointer shape changes before a new window is placed on the screen to indicate to the user that a position needs to be selected for the upper-left corner of the window. If the value is False, windows are placed according to the initial window configuration attributes. The default value of this resource is False.

keyBindings (class **KeyBindings**)
Identifies the set of key bindings for window management functions. If specified, these key bindings replace the built-in default bindings. The named set of key bindings is specified in **mwm** resource description file. The default value for this resource is DefaultKeyBindings.

keyboardFocusPolicy (class **KeyboardFocusPolicy**)
Determines the keyboard focus policy. If set to the pointer value, the keyboard focus policy has the keyboard focus set to the client window that contains the pointer (the pointer could also be in the client window decoration that the **mwm** command adds). If set to the explicit value, the policy is to have the keyboard focus set to a client window when the user presses the left mouse button with the pointer on the client window or any part of the associated **mwm** decoration. The default value for this resource is explicit.

limitResize (class **LimitResize**)
Determines whether the user is allowed to resize a window to greater than the maximum size. If this resource is True, the user is not allowed to resize a window to greater than the maximum size. The default value for this resource is True.

lowerOnIconify (class **LowerOnIconify**)
Determines whether a window icon is displayed on the bottom of the window stack when the window is iconified (minimized). A value of False places the icon in the stacking order at the same place as its associated window. The default value of this resource is True.

maximumMaximumSize (class **MaximumMaximumSize**)
Limits the maximum size of a client window as set by the user or client. The resource value is *Width* x *Height* (for example, 1024x1024) where the width and height are in pixels. The default value of this resource is twice the screen width and height.

moveOpaque (class **MoveOpaque**)
Controls whether the actual window is moved or a rectangular outline of the window is moved. A default value of False displays a rectangular outline on move operations.

moveThreshold (class **MoveThreshold**)
Controls the sensitivity of dragging operations that move windows and icons. The value of this resource is the number of pixels that the locator is moved with a button down before the move operation is initiated. This is used to prevent window and icon movement when you click or double-click and there is unintentional pointer movement with the button down. The default value of this resource is 4 (pixels).

multiScreen (class **MultiScreen**)
Determines whether the **mwm** command manages all the screens on the display. If False, the **mwm** command manages only a single screen. The default value is False.

passButtons (class **PassButtons**)
Indicates whether button press events are passed to clients after they are used to do a window manager function in the client context. If the resource value is False, the button press is not passed to the client. If the value is True, the button press is passed to the client window. The window manager function is done in either case. The default value for this resource is False.

passSelectButton (class **PassSelectButton**)
Indicates whether to pass the select button press events to clients after they are used to do a window manager function in the client context. If the resource value is False, the button press is not passed to the client. If the value is True, the button press is passed to the client window. The window manager function is done in either case. The default value for this resource is True.

positionIsFrame (class **PositionIsFrame**)
Indicates how client window position information (from the **WM_NORMAL_HINTS** property and from configuration requests) is to be interpreted. If the resource value is True, the information is interpreted as the position of the MWM client window frame. If the value is False, it is interpreted as being the position of the client area of the window. The default value of this resource is True.

positionOnScreen (class **PositionOnScreen**)
Indicates that windows should initially be placed (if possible) so that they are not clipped by the edge of the screen (if the resource value is True). If a window is larger than the size of the screen, at least the upper-left corner of the window is on-screen. If the resource value is False, windows are placed in the requested position even if totally off-screen. The default value of this resource is True.

quitTimeout (class **QuitTimeout**)
Specifies the amount of time in milliseconds that the **mwm** command waits for a client to update the **WM_COMMAND** property after the **mwm** command has sent the **WM_SAVE_YOURSELF** message. This protocol is used only for those clients that have a **WM_SAVE_YOURSELF** atom and no **WM_DELETE_WINDOW** atom in the **WM_PROTOCOLS** client window property. The default value of this resource is 1000 (milliseconds). See the **f.kill** function for additional information.

raiseKeyFocus (class **RaiseKeyFocus**)
Specifies whether a window raised by means of the **f.normalize_and_raise** function also receives the input focus. The default value of this resource is False. This resource is available only when the keyboard input focus policy is set to the explicit value.

resizeBorderWidth (class **ResizeBorderWidth**)
Specifies the width (in pixels) of a client window frame border with resize handles. The specified border width includes the 3-D shadows. The default value is based on the size and resolution of the screen.

resizeCursors (class **ResizeCursors**)
Indicates whether the resize cursors are always displayed when the pointer is in the window size border. If True, the cursors are shown; otherwise, the window manager cursor is shown. The default value is True.

screens (class **Screens**)
Specifies the resource names to use for the screens managed by the **mwm** command. If the **mwm** command is managing a single screen, only the first name in the list is used. If the **mwm** command is managing multiple screens, the names are assigned to the screens in order, starting with screen 0. For example, screen 0 gets the first name and screen 1 gets the second name. Examples of default screen names are 0 and 1.

showFeedback (class **ShowFeedback**)

Controls when feedback information is displayed. It controls both window position and size feedback during move or resize operations and initial client placement. It also controls window manager message and dialog boxes.

The value for this resource is a list of names of the feedback options to be enabled or disabled; the names must be separated by a space. If an option is preceded by a minus sign, that option is excluded from the list. The sign of the first item in the list determines the initial set of options. If the sign of the first option is - (a minus sign), the **mwm** command assumes all options are present and starts subtracting from that set. If the sign of the first decoration is + (a plus sign) or is not specified, the **mwm** command starts with no options and builds up a list from the resource.

The names of the feedback options are as follows:

all Shows all feedback (default value).

behavior
Confirms the behavior switch.

kill Confirms on receipt of the **KILL** signal.

move Shows position during the move.

none Shows no feedback.

placement
Shows position and size during initial placement.

quit Confirms quitting MWM.

resize Shows size during resize.

restart Confirms MWM restart.

The following command line illustrates the syntax for the **showFeedback** resource:

Mwm*showFeedback: placement resize behavior restart

This resource specification provides feedback for initial client placement and resize, and it enables the dialog boxes to confirm the restart and set behavior functions. It disables feedback for the move function.

The default value for this resource is the all value.

startupKeyFocus (class **StartupKeyFocus**)

Determines whether a window gets the keyboard input focus when the window is mapped (that is, initially managed by the window manager). It is recommended that both the **autoKeyFocus** resource and the **startupKeyFocus** resource be set to a value of True to work with tear-off menus. The default value is True. This resource is available only when the keyboard input focus policy is set to the explicit value.

transientDecoration (class **TransientDecoration**)
Controls the amount of decoration that Mwm puts on transient windows. The decoration specification is exactly the same as for the **clientDecoration** (client-specific) resource. Transient windows are identified by the **WM_TRANSIENT_FOR** property, which is added by the client to indicate a relatively temporary window. The default value for this resource is the menu title value (that is, transient windows have resize borders and a title bar with a window menu button).

An application can also specify which decorations the window manager should apply to its windows. If it does so, the window manager applies only those decorations indicated by both the application and the **transientDecoration** resource. Otherwise, the window manager applies only the decorations indicated by the **transientDecoration** resource.

transientFunctions (class **TransientFunctions**)
Indicates which window management functions are applicable (or not applicable) to transient windows. The function specification is exactly the same as for the **clientFunctions** (client-specific) resource. The default value for this resource is **-minimize -maximize**.

An application can also specify which functions the window manager should apply to its windows. If it does so, the window manager applies only those functions indicated by both the application and the **transientFunctions** resource. Otherwise, the window manager applies only the functions indicated by the **transientFunctions** resource.

useIconBox (class **UseIconBox**)
Determines whether icons are placed in an icon box. If this resource is given a value of True, icons are placed in an icon box. When an icon box is not used, the icons are placed on the root window (default value).

wMenuButtonClick (class **WMenuButtonClick**)
Indicates whether the window menu is posted and remains posted after a click of the mouse when the pointer is over the **Window Menu** button. If the value given this resource is True, the menu remains posted. True is the default value for this resource.

wMenuButtonClick2 (class **WMenuButtonClick2**)
Indicates whether a double-click action on the **Window Menu** button performs an **f.kill** function. When this resource is given the default value of True, a double-click action on the **Window Menu** button performs an **f.kill** function.

Client-Specific Resources

The syntax for specifying client-specific resources is as follows:

Mwm*ClientNameOrClass***ResourceID*

For example, `Mwm*mterm*windowMenu` is used to specify the window menu to be used with `mterm` clients.

The syntax for specifying client-specific resources for all classes of clients is as follows:

Mwm*ResourceID*

Specific client specifications take precedence over the specifications for all clients. For example, **Mwm*windowMenu** is used to specify the window menu to be used for all classes of clients that do not have a window menu specified.

The syntax for specifying resource values for windows that have an unknown name and class (that is, windows that do not have a **WM_CLASS** property associated with them) is as follows:

Mwm*defaults*ResourceID*

For example, **Mwm*defaults*iconImage** is used to specify the icon image to be used for windows that have an unknown name and class.

Client-Specific Resource Set

Note: Hyphens in the following table are for readability purposes only. Do not include hyphens within names in programs.

Name	Class	Value Type	Default
clientDecoration	**ClientDecoration**	all	
clientFunctions	**ClientFunctions**	string	all
focusAutoRaise	**FocusAutoRaise**	True or False	varies
iconImage	**IconImage**	pathname	(image)
iconImageBackground	**Background**	color	icon background
iconImageBottomShadowColor	**Foreground**	color	icon bottom shadow
iconImageBottomShadowPixmap	**BottomShadowPixmap**	color	icon bottom shadow pixmap
iconImageForeground	**Foreground**	color	varies
iconImageTopShadowColor	**Background**	color	icon top shadow color
iconImageTopShadoPixmap	**TopShadowPixmap**	color	icon top shadow pixmap
matteBackground	**Background**	color	background
matteBottomShadowColor	**Foreground**	color	bottom shadow color
matteBottomShadowPixmap	**BottomShadowPixmap**	color	bottom shadow pixmap
matteForeground	**Foreground**	color	foreground
matteTopShadowColor	**Background**	color	top shadow color
matteTopShadowPixmap	**TopShadowPixmap**	color	top shadow pixmap
matteWidth	**MatteWidth**	pixels	0
maximumClientSize	**MaximumClientSize**	wxh, vertical, horizontal	fill the screen
useClientIcon	**UseClientIcon**	True or False	F
usePPosition	**UsePPosition**	string	nonzero
windowMenu	**WindowMenu**	string	DefaultWindowMenu

clientDecoration (class **ClientDecoration**)

Controls the amount of window frame decoration. The resource is specified as a list of decorations to specify their inclusion in the frame. If a decoration is preceded by - (a minus sign), that decoration is excluded from the frame. The sign of the first item in the list determines the initial amount of decoration. If the sign of the first decoration is a minus sign, the **mwm** command assumes all decorations are present and starts subtracting from that set. If the sign of the first decoration is plus (or not specified), the **mwm** command starts with no decoration and builds up a list from the resource.

An application can also specify which decorations the **mwm** command should apply to its windows. If it does so, the **mwm** command applies only those decorations indicated by both the application and the **clientDecoration** resource. Otherwise, the **mwm** command applies the decorations indicated by the **clientDecoration** resource. Following is a list of window frame decorations:

all Specifies to include all decorations (default value).

border Specifies the window border.

maximize
Specifies the Maximize button (includes title bar).

minimize
Specifies the Minimize button (includes title bar).

none Specifies no decorations.

resizeh
Specifies the border resize handles (includes border).

menu Specifies the Window Menu button (includes title bar).

title Specifies the title bar (includes border).

Following are examples of window frame decoration commands:

```
Mwm*XClock.clientDecoration: -resizeh -maximize
```

This removes the resize handles and Maximize button from XClock windows.

```
Mwm*XClock.clientDecoration: menu minimize border
```

This removes the resize handles and Maximize button from XClock windows. Note that either `menu` or `minimize` implies title.

clientFunctions (class **ClientFunctions**)

Indicates which **mwm** functions are applicable (or not applicable) to the client window. The value for the resource is a list of functions. If the first function in the list has - (a minus sign) in front of it, the **mwm** command starts with all functions and subtracts from that set. If the first function in the list has a + (plus sign) in front of it, the **mwm** command starts with no functions and builds up a list. Each function in the list must be preceded by the appropriate + (plus) or - (minus) sign and separated from the next function by a space.

An application can also specify which functions the **mwm** command should apply to its windows. If it does so, the **mwm** command applies only those functions indicated by both the application and the **clientFunctions** resource. Otherwise, the **mwm** command applies the functions indicated by the **clientFunctions** resource.

Following is a list of functions available for this resource:

all Specifies to include all functions (default value).

none Specifies no functions.

resize Specifies **f.resize**.

move Specifies **f.move**.

minimize
Specifies **f.minimize**.

maximize
Specifies **f.maximize**.

close Specifies **f.kill**.

focusAutoRaise (class **FocusAutoRaise**)

Determines whether clients are raised when they get the keyboard input focus. If the value is False, the stacking of windows on the display is not changed when a window gets the keyboard input focus. The default value is True when the **keyboardFocusPolicy** is the explicit value and False when the **keyboardFocusPolicy** is the pointer value.

iconImage (class **IconImage**)

Specifies an icon image for a client (for example, **Mwm*myclock*iconImage**). The resource value is a path name for a bitmap file. The value of the (client-specific) **useClientIcon** resource is used to determine whether user-supplied icon images are used instead of client-supplied icon images. The default value is to display a built-in window manager icon image.

iconImageBackground (class **Background**)

Specifies the background color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon background color (that is, specified by **Mwm*background** or **Mwm*icon*background**).

iconImageBottomShadowColor (class **Foreground**)

Specifies the bottom shadow color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon bottom shadow color (that is, specified by **Mwm*icon*bottomShadowColor**).

iconImageBottomShadowPixmap (class **BottomShadowPixmap**)

Specifies the bottom shadow pixmap of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon bottom shadow pixmap (that is, specified by **Mwm*icon*bottomShadowPixmap**).

iconImageForeground (class **Foreground**)
Specifies the foreground color of the icon image that is displayed in the image part of an icon. The default value of this resource varies depending on the icon background.

iconImageTopShadowColor (class **Background**)
Specifies the top shadow color of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon top shadow color (that is, specified by **Mwm*icon*topShadowColor**).

iconImageTopShadowPixmap (class **TopShadowPixmap**)
Specifies the top shadow pixmap of the icon image that is displayed in the image part of an icon. The default value of this resource is the icon top shadow pixmap (that is, specified by **Mwm*icon*topShadowPixmap**).

matteBackground (class **Background**)
Specifies the background color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client background color (that is, specified by **Mwm*background** or **Mwm*client*background**).

matteBottomShadowColor (class **Foreground**)
Specifies the bottom shadow color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client bottom shadow color (that is, specified by **Mwm*bottomShadowColor** or **Mwm*client*bottomShadowColor**).

matteBottomShadowPixmap (class **BottomShadowPixmap**)
Specifies the bottom shadow pixmap of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client bottom shadow pixmap (that is, specified by **Mwm*bottomShadowPixmap** or **Mwm*client*bottomShadowPixmap**).

matteForeground (class **Foreground**)
Specifies the foreground color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client foreground color (that is, specified by **Mwm*foreground** or **Mwm*client*foreground**).

matteTopShadowColor (class **Background**)
Specifies the top shadow color of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client top shadow color (that is, specified by **Mwm*topShadowColor** or **Mwm*client*topShadowColor**).

matteTopShadowPixmap (class **TopShadowPixmap**)
Specifies the top shadow pixmap of the matte when the **matteWidth** resource is a positive value. The default value of this resource is the client top shadow pixmap (that is, specified by **Mwm*topShadowPixmap** or **Mwm*client*topShadowPixmap**).

matteWidth (class **MatteWidth**)
Specifies the width of the optional matte. The default value is 0, which effectively disables the matte.

maximumClientSize (class **MaximumClientSize**)
Indicates the client size to be used when an application is maximized. The resource value is specified *Width*x*Height*. The width and height are interpreted in the units that the client uses (for example, this is generally characters for terminal emulators). Alternately, the vertical or horizontal value can be specified to indicate the direction in which the client maximizes. If this resource is not specified, the maximum size from the **WM_NORMAL_HINTS** property is used if set. Otherwise, the default value is the size where the client window with window management borders fills the screen. When the maximum client size is not determined by the **maximumClientSize** resource, the **maximumMaximumSize** resource value is used as a constraint on the maximum size.

useClientIcon (class **UseClientIcon**)
Determines whether a client-supplied icon image takes precedence over a user-supplied icon image. The default value is False, giving the user-supplied icon image higher precedence than the client-supplied icon image.

usePPosition (class **UsePPosition**)
Specifies whether the window manager honors the program-specified position **PPosition** specified in the **WM_NORMAL_HINTS** property in the absence of a user-specified position. Setting this resource to On causes the **mwm** command to always honor the program-specified position. Setting this resource to Off causes the **mwm** command to always ignore the program-specified position. Setting this resource to the default value of nonzero causes the **mwm** command to honor program-specified positions other than (0,0).

windowMenu (class **WindowMenu**)
Indicates the name of the menu pane that is posted when the window menu is opened (usually by pressing button 1 on the Window Menu button on the client window frame). Menu panes are specified in the **mwm** resource description file. Window menus can be customized on a client class basis by specifying resources of the form **Mwm***_ClientNameOrClass_***windowMenu** (See mwm Resource Description File Syntax for more information.) The default value of this resource is DefaultWindowMenu.

Resource Description File

The **mwm** resource description file is a supplementary resource file that contains resource descriptions that are referred to by entries in the defaults files (**.Xdefaults**, **app-defaults/Mwm**). It contains descriptions of resources that are to be used by the **mwm** command and that cannot be easily encoded in the defaults files (a bitmap file is an analogous type of resource description file). A particular **mwm** resource description file can be selected using the **configFile** resource.

The following types of resources can be described in the **mwm** resource description file:

buttons Window manager functions can be bound (associated) with button events.
keys Window manager functions can be bound (associated) with key press events.
menus Menu panes can be used for the window menu and other menus posted with key bindings and button bindings.

mwm Resource Description File Syntax

The **mwm** resource description file is a standard text file that contains items of information separated by blanks, tabs, and new-line characters. Blank lines are ignored. Items or characters can be quoted to avoid special interpretation (for example, the # (comment character) can be quoted to prevent it from being interpreted as the comment character). A quoted item can be contained in " " (double quotation marks). Single characters can be quoted by preceding them with the \ (backslash). All text from an unquoted # (comment character) to the end of the line is regarded as a comment and is not interpreted as part of a resource description. If an ! (exclamation mark) is the first character in a line, the line is regarded as a comment. If a line ends in a \ (backslash), the next line is considered a continuation of that line.

Window manager functions can be accessed with button and key bindings and with window manager menus. Functions are indicated as part of the specifications for button and key binding sets and for menu panes. The function specification has the following syntax:

```
Function = FunctionName [FunctionArguments]
FunctionName = Window Manager Function
FunctionArguments = {QuotedItem | UnquotedItem}
```

The following functions are supported. If a function is specified that is not one of the supported functions, it is interpreted by the **mwm** command as the **f.nop** function.

f.beep	Causes a beep.
f.circle_down [*Icon* | *Window*]	Causes the window or icon that is on the top of the window stack to be put on the bottom of the window stack (so that it no longer obscures any other window or icon). This function affects only those windows and icons that obscure other windows and icons or that are obscured by other windows and icons. Secondary windows (that is, transient windows) are restacked with their associated primary window. Secondary windows always stay on top of the associated primary window and there can be no other primary windows between the secondary windows and their primary window. If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.circle_up [*Icon* | *Window*]	Raises the window or icon on the bottom of the window stack (so that it is not obscured by any other windows). This function affects only those windows and icons that obscure other windows and icons or that are obscured by other windows and icons. Secondary windows (that is, transient windows) are restacked with their associated primary window. If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.exec or **!**	Causes the command to be run (using the value of the **MWMSHELL** environment variable if it is set; otherwise, the value of the **SHELL** environment variable if it is set; otherwise, **/usr/bin/sh** is used). The ! notation can be used in place of the **f.exec** function name.
f.focus_color	Sets the colormap focus to a client window. If this function is done in a root context, the default colormap (set up by the X Window System client for the screen where MWM is running) is installed and there is no specific client window colormap focus. This function is treated as **f.nop** if **colormapFocusPolicy** is not set to the explicit value.
f.focus_key	Sets the keyboard input focus to a client window or icon. This function is treated as **f.nop** if **keyboardFocusPolicy** is not set to the explicit value or the function is run in a root context.
f.kill	Stops a client. If the **WM_DELETE_WINDOW** protocol is set up, the client is sent a client message event indicating that the client window needs to be deleted. If the **WM_SAVE_YOURSELF** protocol is set up and the **WM_DELETE_WINDOW** protocol is not set up, the client is sent a client message event indicating that the client needs to prepare to be stopped. If the client does not have the **WM_DELETE_WINDOW** or **WM_SAVE_YOURSELF** protocol set up, this function causes a client's X connection to be stopped (usually resulting in the end of the client). See the description of the **quitTimeout** resource.

f.lower [*-Client* | **within** | **freeFamily**]	Lowers a client window to the bottom of the window stack (where it obscures no other window). Secondary windows (that is, transient windows) are restacked with their associated primary window. The *Client* argument indicates the name or class of a client to lower. If the *Client* argument is not specified, the context, in which the function was started, indicates the window or icon to lower.
f.maximize	Causes a client window to be displayed with its maximum size.
f.menu	Associates a cascading (pull-right) menu with a menu pane entry or a menu with a button or key binding. The **menu_name** function argument identifies the menu to be used.
f.minimize	Causes a client window to be iconified (minimized). When a window is minimized and no icon box is used, its icon is placed on the bottom of the window stack (so that it obscures no other windows). If an icon box is used, the client's icon changes to its iconified form inside the icon box. Secondary windows (that is, transient windows) are minimized with their associated primary window. There is only one icon for a primary window and all its secondary windows.
f.move	Causes a client window to be interactively moved.
f.next_cmap	Installs the next colormap in the list of colormaps for the window with the colormap focus.
f.next_key [*Icon* | *Window* | *Transient*]	Sets the keyboard input focus to the next window or icon in the set of windows and icons managed by the window manager (the ordering of this set is based on the stacking of windows on the screen). This function is treated as **f.nop** if **keyboardFocusPolicy** is not the explicit value. The keyboard input focus is moved only to windows that do not have an associated secondary window that is application-modal. If the *Transient* argument is specified, transient (secondary) windows are crossed (otherwise, if only the *Window* argument is specified, traversal is done only to the last focused window in a transient group). If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.
f.nop	Does nothing. If a function is specified in a type of resource where it is not supported or is started in a context that does not apply, the function is treated as **f.nop**.
f.normalize	Causes a client window to be displayed with its normal size. Secondary windows (that is, transient windows) are placed in their normal state along with their associated primary window.
f.normalize_and_raise	Causes the corresponding client window to be displayed with its normal size and raised to the top of the window stack. Secondary windows (that is, transient windows) are placed in their normal state along with their associated primary window.
f.pack_icons	Causes icons to be packed into the icon grid. This function is used to relay out icons (based on the layout policy being used) on the root window or in the icon box.

f.pass_keys
Enables or disables (toggles) processing of key bindings for window manager functions. When it disables key binding processing, all keys are passed on to the window with the keyboard input focus and no window manager functions are started. If the **f.pass_keys** function is started with a key binding to disable key-binding processing, the same key binding can be used to enable key-binding processing.

f.post_wmenu
Posts the window menu. If a key is used to post the window menu and the Window Menu button is present, the window menu is automatically placed with its top-left corner at the bottom-left corner of the Window Menu button for the client window. If no Window Menu button is present, the window menu is placed at the top-left corner of the client window.

f.prev_cmap
Installs the previous colormap in the list of colormaps for the window with the colormap focus.

f.prev_key [*Icon* | *Window* | *Transient*]
Sets the keyboard input focus to the previous window or icon in the set of windows and icons managed by the window manager (the ordering of this set is based on the stacking of windows on the screen). This function is treated as **f.nop** if **keyboardFocusPolicy** is not the explicit value. The keyboard input focus is moved only to windows that do not have an associated secondary window that is application-modal. If the *Transient* argument is specified, transient (secondary) windows are crossed (otherwise, if only window is specified, traversal is done only to the last focused window in a transient group). If an *Icon* function argument is specified, the function applies only to icons. If a *Window* function argument is specified, the function applies only to windows.

f.quit_mwm
Stops the **mwm** command (but *not* the X Window System client).

f.raise [*-Client* | **within** | **freeFamily**]
Raises a client window to the top of the window stack (where it is obscured by no other window). Raises the secondary window (transient window or dialog box) within the client family. The arguments to this function are mutually exclusive. The *Client* argument indicates the name or class of a client to raise. If the *Client* argument is not specified, the context in which the function was started indicates the window or icon to raise. Specifying **within** raises the secondary window within the family but does not raise the client family in the global window stack. Specifying **freeFamily** raises the window to the top of its local family stack and raises the family to the top of the global window stack.

f.raise_lower [**within** | **freeFamily**]

Raises a primary window to the top of the window stack if it is partially obscured by another window; otherwise, it lowers the window to the bottom of the window stack. The arguments to this function are mutually exclusive.

Specifying **within** raises a secondary window within the family (staying above the parent window), if it is partially obscured by another window in the application's family; otherwise, it lowers the window to the bottom of the family stack. It has no effect on the global stacking order.

Specifying **freeFamily** raises the window to the top of its local family stack, if obscured by another window, and raises the family to the top of the global window stack; otherwise, it lowers the window to the bottom of its local family stack and lowers the family to the bottom of the global window stack.

f.refresh

Causes all windows to be redrawn.

f.refresh_win

Causes a client window to be redrawn.

f.resize

Causes a client window to be interactively resized.

f.restart

Causes the **mwm** command to be restarted (effectively stopped and restarted).

f.restore

Restores the previous state of an icon's associated window. If a maximized window is iconified, the **f.restore** function restores it to its maximized state. If a normalized window is iconified, the **f.restore** function restores it to its normalized state.

f.restore_and_raise

Restores the previous state of an icon's associated window and raises the window to the top of the window stack. If a maximized window is iconified, the **f.restore_and_raise** function restores it to its maximized state and raises it to the top of the window stack. If a normalized window is iconified, the **f.restore_and_raise** function restores it to its normalized state and raises it to the top of the window stack.

f.screen [**next** | **prev** | **back** | *ScreenNumber*]

Causes the pointer to warp to a specific screen number or to the next, previous, or last visited screen. The arguments to this function are mutually exclusive. The following arguments are available:

ScreenNumber
: Indicates the screen number to which the pointer is warped. Screens are numbered starting from screen 0.

next
: Warps the pointer to the next managed screen (skipping over any unmanaged screens).

prev
: Warps the pointer to the previous managed screen (skipping over any unmanaged screens).

back
: Warps the pointer to the last visited screen.

f.send_msg *MessageNumber*

Sends a client message of the **_MOTIF_WM_MESSAGES** type with the *MessageType* indicated by the *MessageNumber* function argument. The client message is sent only if *MessageNumber* is included in the client's **_MOTIF_WM_MESSAGES** property. A menu item label is unavailable if the menu item is used to perform the **f.send_msg** function of a message that is not included in the client's **_MOTIF_WM_MESSAGES** property.

f.separator

Causes a menu separator to be put in the menu pane at the specified location (the label is ignored).

f.set_behavior

Causes the window manager to restart with the default behavior (if a custom behavior is configured) or revert to the custom behavior. By default this is bound to the Shift+Ctrl+Meta+! key sequence.

The Meta+Shift+Ctrl+! key sequence switches (that is, toggles) between the default and custom behaviors. When the user switches to the default MWM behavior, a number of **mwm** resources assume their default values and the **mwm** command restarts. When the user switches back to the custom behavior, the resource values that were changed to default values are reset with the custom values and the **mwm** command restarts.

When an **f.set_behavior** function is performed, the following user interaction occurs:

1. A system-modal dialog box is displayed prompting the user for confirmation of the **f.set_behavior** action.
2. The user can cancel the action at this point.
3. The window manager restarts.
4. The window manager applies the new (custom or default) configuration values.
5. Window manager components are mapped.

When the default MWM behavior is being set, default resource values are applied and, if specified, client properties that control window manager behavior are applied. This includes the **_MOTIF_WM_HINTS** and **_MOTIF_WM_MENU** properties. These properties might alter default MWM behavior, but it is done in a way that is consistent for all users.

f.title

Inserts a title in the menu pane at the specified location.

Function Contexts

Each function may be constrained as to which resource types can specify the function (for example, menu pane) and also what context the function can be used in (for example, the function is done to the selected client window). The following are the function contexts:

root No client window or icon is selected as an object for the function.

window A client window is selected as an object for the function. This includes the window's title bar and frame. Some functions are applied only when the window is in its normalized state (for example, **f.maximize**) or its maximized state (for example, **f.normalize**).

icon An icon is selected as an object for the function.

If a function's context is specified as **icon**|**window** and the function is started in an icon box, the function applies to the icon box, not to the icons inside.

If a function is specified in a type of resource where it is not supported or is started in a context that does not apply, the function is treated as **f.nop**. The following table indicates the resource types and function contexts in which window manager functions apply:

Function	Contexts	Resources
f.beep	root, icon, window	button, key, menu
f.circle_down	root, icon, window	button, key, menu
f.circle_up	root, icon, window	button, key, menu

Function	Contexts	Resources
f.exec	root, icon, window	button, key, menu
f.focus_color	root, icon, window	button, key, menu
f.focus_key	root, icon, window	button, key, menu
f.kill	icon, window	button, key, menu
f.lower	icon, window	button, key, menu
f.maximize	icon, window (normal)	button, key, menu
f.menu	root, icon, window	button, key, menu
f.minimize	window	button, key, menu
f.move	icon, window	button, key, menu
f.next_cmap	root, icon, window	button, key, menu
f.next_key	root, icon, window	button, key, menu
f.nop	root, icon, window	button, key, menu
f.normalize	icon, window (maximized)	button, key, menu
f.normalize_and_raise	icon, window	button, key, menu
f.pack_icons	root, icon, window	button, key, menu
f.pass_keys	root, icon, window	button, key, menu
f.post_wmenu	root, icon, window	button, key
f.prev_cmap	root, icon, window	button, key, menu
f.prev_key	root, icon, window	button, key, menu
f.quit_mwm	root, icon, window	button, key, menu (root only)
f.raise	icon, window	button, key, menu
f.raise_lower	icon, window	button, key, menu
f.refresh	root, icon, window	button, key, menu
f.refresh_win	window	button, key, menu
f.resize	window	button, key, menu
f.restart	root, icon, window	button, key, menu
f.restore	icon, window	button, key, menu
f.restore_and_raise	icon, window	button, key, menu
f.screen	root, icon, window	button, key, menu
f.send_msg	icon, window	button, key, menu
f.separator	root, icon, window	menu
f.set_behavior	root, icon, window	button, key, menu
f.title	root, icon, window	menu

Window Manager Event Specification

Events are indicated as part of the specifications for button and key-binding sets and for menu panes.

Button events have the following syntax:

Button = [*ModifierList*]<*ButtonEventName*>
ModifierList = *Modifier Name* {*ModifierName*}

All modifiers specified are interpreted as being exclusive (this means that only the specified modifiers can be present when the button event occurs). Following is a list that indicates the values that can be used for the *ModifierName* parameter. The Alt key is frequently labeled Extend or Meta. Alt and Meta can be used interchangeably in event specification.

Modifier	Description
Ctrl	Control key
Shift	Shift key
Alt	Alt or Meta key
Meta	Meta or Alt key
Lock	Lock key
Mod1	Modifier1
Mod2	Modifier2
Mod3	Modifier3
Mod4	Modifier4
Mod5	Modifier5

Following is a list that indicates the values that can be used for the *ButtonEventName* parameter.

Button	Description
Btn1Down	Button 1 press
Btn1Up	Button 1 release
Btn1Click	Button 1 press and release
Btn1Click2	Button 1 double click
Btn2Down	Button 2 press
Btn2Up	Button 2 release
Btn2Click	Button 2 press and release
Btn2Click2	Button 2 double click
Btn3Down	Button 3 press
Btn3Up	Button 3 release
Btn3Click	Button 3 press and release
Btn3Click2	Button 3 double click
Btn4Down	Button 4 press
Btn4Up	Button 4 release
Btn4Click	Button 4 press and release
Btn4Click2	Button 4 double click
Btn5Down	Button 5 press
Btn5Up	Button 5 release
Btn5Click	Button 5 press and release
Btn5Click2	Button 5 double click.

Key events that are used by the window manager for menu mnemonics and for binding to window manager functions are single key presses; key releases are ignored. Key events have the following syntax:

Key = [*ModifierList*] **<Key>** *KeyName*
ModifierList = *ModifierName* {*ModifierName*}

All modifiers specified are interpreted as being exclusive (this means that only the specified modifiers can be present when the key event occurs). Modifiers for keys are the same as those that apply to buttons. The *KeyName* parameter is an X11 keysym name. Key symbol names can be found in the **keysymdef.h** file (remove the *XK_* prefix).

The key symbol names will be resolved to a single specific key code by the Window Manager during startup and will not change unless the Window Manager is restarted.

Button Bindings

The **buttonBindings** resource value is the name of a set of button bindings that are used to configure window manager behavior. A window manager function can be used when a button press occurs with the pointer over a framed client window, an icon, or the root window. The context for indicating where the button press applies is also the context for starting the window manager function when the button press is done (significant for functions that are context-sensitive).

Following is the button binding syntax:

```
Buttons BindingsSetName
        {
Button Context Function
Button Context Function
                .
                .
Button Context Function
        }
```

Following is the syntax for the context specification:

Context = *Object*[|*Context*]
Object = **root** | **icon** | **window** | **title** | **frame** | **border** | **app**

The *Context* specification indicates where the pointer must be for the button binding to be effective. For example, a context of **window** indicates that the pointer must be over a client window or window management frame for the button binding to be effective. The **frame** context is for the window management frame around a client window (including the border and title bar), the **border** context is for the border part of the window management frame (not including the title bar), the **title** context is for the title area of the window management frame, and the **app** context is for the application window (not including the window management frame).

If an **f.nop** function is specified for a button binding, the button binding is not done.

Key Bindings

The **keyBindings** resource value is the name of a set of key bindings that are used to configure window manager behavior. A window manager function can be done when a particular key is pressed. The context in which the key binding applies is indicated in the key binding specification. The valid contexts are the same as those that apply to button bindings.

Following is the key binding syntax:

```
Keys BindingsSetName
{
Key Context Function
Key Context Function
                .
                .
Key Context Function
}
```

If an **f.nop** function is specified for a key binding, the key binding is not done. If an **f.post_wmenu** or **f.menu** function is bound to a key, the **mwm** command automatically uses the same key for removing the menu from the screen after it is open.

The *Context* specification syntax is the same as for button bindings. For key bindings, the **frame**, **title**, **border**, and **app** contexts are equivalent to the **window** context. The context for a key event is the window or icon that has the keyboard input focus (**root** if no window or icon has the keyboard input focus).

Menu Panes

Menus can be opened using the **f.post_wmenu** and **f.menu** window manager functions. The context for window manager functions that are done from a menu is **root**, **icon**, or **window**, depending on how the menu is opened. In the case of the window menu or menus opened with a key binding, the location of the keyboard input focus indicates the context. For menus opened using a button binding, the context of the button binding is the context of the menu.

Following is the menu pane specification syntax:

```
Menu MenuName
{
Label [Mnemonic] [Accelerator] Function
Label [Mnemonic] [Accelerator] Function
                 .
                 .
Label [Mnemonic] [Accelerator] Function
}
```

Each line in the *Menu* specification identifies the label for a menu item and the function to be completed if the menu item is selected. Optionally, a menu button mnemonic and a menu button keyboard accelerator can be specified. Mnemonics are functional only when the menu is posted and keyboard traversal applies.

The label can be a string or a bitmap file. The *Label* specification has the following syntax:

```
Label = Text | BitmapFile
BitmapFile = @FileName
Text = QuotedItem | UnquotedItem
```

The string encoding for labels must be compatible with the menu font that is used. Labels are not available for menu items that use the **f.nop** function, an invalid function, or a function that does not apply in the current context.

A *Mnemonic* specification has the following syntax:

```
Mnemonic = _Character
```

The first matching *Character* in the label is underlined. If there is no matching *Character* in the label, no mnemonic is registered with the window manager for that label. Although the *Character* must exactly match a character in the label, the mnemonic does not perform if any modifier (such as the Shift key) is pressed with the character key.

The *Accelerator* specification is a key event specification with the same syntax that is used for key bindings to window manager functions.

Environment

The **mwm** command does the following:

- Uses the **HOME** environment variable to specify the user's home directory.
- Uses the **LANG** environment variable to specify the user's choice of language for the **mwm** message catalog and the **mwm** resource description file.
- Uses the **XFILESEARCHPATH**, **XUSERFILESEARCHPATH**, **XAPPLRESDIR**, **XENVIRONMENT**, **LANG**, and **HOME** environment variables to determine search paths for resource defaults files. The **mwm** command can also use the **XBMLANGPATH** environment variable to search for bitmap files.
- Reads the **$HOME/.motifbind** file, if it exists, to install a virtual key bindings property on the root window.
- Uses the **MWMSHELL** environment variable (or **SHELL** if **MWMSHELL** is not set) to specify the shell to use when running commands through the **f.exec** function.

Exit Status

This command returns the following exit values:

0 Indicates successful completion.
>1 Indicates an error occurred.

Files

/usr/lib/X11/$LANG/system.mwmrc

/usr/lib/X11/system.mwmrc

/usr/lib/X11/app-defaults/Mwm

$HOME/Mwm

$HOME/.Xdefaults

$HOME/$LANG/.mwmrc

$HOME/.mwmrc

$HOME/.motifbind

Related Information

The **X** command in *AIX 5L Version 5.2 Commands Reference*.

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, 2003

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

IBM

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be the trademarks or service marks of others.

Index

Special characters

/etc/filesystems file
  listing entries
    using lsfs command 327
/etc/inittab file
  listing records
    using lsitab command 330
  making records
    using mkitab command 515
/etc/rc.nfs
  uncomments entry in
    using mkclient command 489
/etc/vfs file
  listing entries
    using lsvfs command 402

A

account, user
  customizing
    using mkuser.sys command 600
accounting system
  displaying information about last commands run 152
  displaying previous logins and logouts 151
  performing periodic accounting
    using monacct command 614
  updating last log in records 154
acct/* commands
  lastlogin 154
  monacct 614
adapters and devices
  displaying
    using lsmcode command 340
add, command in lb_admin 155
address maps
  in ld command 176
AIXwindows Window Manager
  running
    using mwm command 653
  window management
    using mwm command 653
answering client requests
  for file system mounts
    using mountd daemon 633
appearance
  general
    using mwm command 662
appearance of window
  managing
    using mwm command 653
Appearance resource set
  using mwm command 662
attributes
  displaying characteristics of
    using lsattr command 289
attributes *(continued)*
  displaying possible values of
    using lsattr command 289
  of exported symbols
    in ld command 175

B

behavior
  general
    using mwm command 662
behavior of window
  managing
    using mwm command 653
Behavior resource set
  using mwm command 662
bibliography
  building inverted index for 36
  finding references in
    using lookbib command 223
binder
  using ld command 165
binding
  button
    using mwm command 686
  key
    using mwm command 687
boot image
  creating
    using mkboot command 476
boot record
  creating
    using mkboot command 476
button
  binding
    using mwm command 687

C

C programming language
  checking source code for problems 193
  reformatting programs 32
CacheFS directory
  mounting a
    using mkcfsmnt command 486
CD-ROM
  reporting statistics 96
clean, command in lb_admin 155
Client-Specific resource set
  using mwm command 675
code page
  converting one page encoding set to another 5
command
  mwm 653
commands
  elogevent 212

© Copyright IBM Corp. 1997, 2003

commands *(continued)*
event response resource manager (ERRM)
elogevent 212
logevent 212
ikedb 21
installms 60
installp 50
invscoutd 80
join 114
kdb 118
keycomp 127
keyenvoy 130
ld 160
logevent 212
lppchk 256
lpr 263
ls 278
lsactdef 283
lsattr 289
lsaudrec 294
lscomg 303
lscondition 306
lscondresp 310
lscons 316
lslpp 332
lslv 336
lsmcode 340
lsresponse 368
lsrpdomain 373
lsrpnode 376
lsrsrc 379
lsrsrcdef 384
lssensor 393
mail 422
Mail 422
mailq 434
mailx 422
make 437
mgmtsvr 459
mkboot 476
mkcfsmnt 486
mkclient 489
mkcomg 490
mkcondition 493
mkcondresp 498
mkfs 509
mkitab 515
mklv 523
mknfs 535
mknfsmnt 537
mkprtsv 553
mkresponse 563
mkrole 567
mkrpdomain 569
mkrsrc 572
mksensor 583
mkssys 587
mkstr 590
mkuser 597
mkuser.sys 600
mkvg 601
commands, mtrace 646
component
appearance resources
using mwm command 658
Component Appearance resource set
using mwm command 659
computer languages
C
lexical analyzer 184
configuring system
to run NFS
using mknfs command 535
connections
displaying for a device
using lsconn command 314
console device
writing name to standard output
using lscons command 316
CPU
statistics 96
creating
boot image, boot record, and service record
using mkboot command 476
error message file
using mkstr command 590
file system
using mkfs command 509
logical volume
using mklv command 523
new user account
using mkuser command 597
volume group
using mkvg command 601
customizing
new user account
using mkuser.sys command 600

D

daemon
launching Inventory Scout server
using invscoutd command 80
daemons
llbd 201
lockd 209
monitord 616
mountd 633
mrouted 638
datagram, multicast
forwarding a
using mrouted daemaon 638
default window menu
using mwm command
using mwm command 654
delete, command in lb_admin 156
description file
creating
using makedev command 447
using mwm command 679

device
adding to the system
using mkdev command 500
Device Configuration Database 350
listing acceptable devices from
using lsparent command 350
device stanza name
displaying the
using lsquedev command 364
devices
displaying characteristics of
using lsdev command 317
devices and adapters
displaying
using lsmcode command 340
diagnostic messages
issuing by optional programs 77
directories
creating 503
creating a hierarchy 504
displaying contents
using ls command 278
moving 648, 650
renaming 650
directory
creating a lost and found
using mklost+found command 523
mounting
using mount command 624
using infocntr command 42
disks
reporting statistics 96
displaying
directory contents
using ls command 278
displaying system images
for examining a dump
using kdb command 118
displays
files
one screen at a time 618
listing currently available on the system
using lsdisp command 324
document
typesetting
using mmt command 609
dump, examining
displaying system images
using kdb command 118

E

elogevent command 212
elogevent script 212
encryption
generating an encryption key
using makekey command 447
enqueuing
print jobs
using lpr command 263
environment variable
using mwm command 688
environment variables
in ld command 177
ERRM
event information
logging 212
ERRM commands
elogevent 212
logevent 212
ERRM scripts
elogevent 212
logevent 212
error message file
creation
using mkstr command 590
error messages
issuing by optional programs 76
event information
logging 212
event response resource manager (ERRM)
commands
elogevent 212
logevent 212
event information
logging 212
scripts
elogevent 212
logevent 212
event specification
using mwm command 685

F

fields, data
joining from two files
using join command 114
file
creating a special
using mknod command 540
finding lines in a sorted
using look command 222
resource description
using mwm command 679
writing inode information 111
file system
constructing a prototype
using mkproto command 546
making
using mkfs command 509
making available for use
using mount command 624
file system mounts
answering client requests
using mountd daemon 633
files
archive
using ld command 161
creating links 202
displaying
one screen at a time 618

files *(continued)*
import and export
using ld command 161
joining the data fields of two
using join command 114
linking object
using ld command 160
moving 648
filesets
installing, associated with keywords or fixes 63
fixes
installing filesets associated with 63
flags
ignored and unsupported
using ld command 163
folder
incorporating new mail into 30
folders
printing full path names of
using mhpath command 464
font code
adding to the system
using mkfont command 507
font files
creating fonts.dir file
using mkfontdir command 507
font set
changing
using managefonts command 452
updating
using managefonts command 452
fonts
listing the fonts available for use
using lsfont command 326
Frame Component resource set
using mwm command 661

G

games
number-guessing game 617
garbage collection
using ld command 162
global location broker
administrative tools for 154
groups
creating new groups
using mkgroup command 511
displaying attributes of
using lsgroup command 328

H

help
using files, editors, and macros 180
hlpid 6
hlpindent 32
hlpiostat 96
hlpipcs 101
hlplearn 180
hlpleave 181
hlpline 191
hlplint 193
hlpln 202
hlplorder 224
hlpmkvgdata 604

I

I/O
reporting statistics 96
IBM 3812 Model 2 Pageprinter
postprocessing troff command output for 1
IBM 3816 Pageprinter
postprocessing troff command output for 2
ibm3812 command 1
ibm3816 command 2
ibm5587G command 4
IBM5587G printer
postprocessing troff command output for 4
ibm558H-T Command 3
icon
box menu
using mwm command 656
putting in icon box
using mwm command 656
window menu
using mwm command 655
Icon Component resource set
using mwm command 661
iconifying window
using mwm command 655
iconv command 5
id command 6
idinstal 43
idnls 204
idprocess 100
ifconfig command 8
IKE database
retrieves, updates, deletes, imports, and exports information
using ikedb command 21
ikedb command 21
IMAP commands
imapd 25
imapd daemon 25
impfilt command 26
import and export file format
-bI: and -bE: Flags
using ld command 173
importing filter rules
from export files 26
importvg command 27
imptun command
adding exported tunnel definitions and filter rules 29
inc command 30
indent command 32
index
building inverted for bibliography 36
indxbib command 36
inetd daemon 37

infocmp command
managing terminfo descriptions 39
infocntr command 42
inode numbers
information about 111
input extension records
listing
using listX11input command 201
input focus
keyboard
using mwm command 656
input method keymap file
compiling a keyboard mapping file into an
using keycomp command 127
install applications
Installation Assistant 47
install command 43
install command (BSD) 49
install procedure
saving files changed during 73
install_all_updates command 45
install_assist command 47
install_mh command 48
installable software product
verify files
using lppchk command 256
Installation Assistant
starting
using the install_assist command 47
installing
programs in compatible package
using installp command 50
installing a command 43
installms command 60
installp command 50
installp format
creating software packages 517
instfix command 63
intermediary between user processes
keyserv daemon
using keyenvoy command 130
interprocess communication
removing identifiers 100
reporting status 101
inucp command 65
inudocm command 66
inurecv command
description of 69
performing archive operations for 71
performing restore operations for 71
inurest command 70
inurid command
removing installation information 72
inusave command
description of 73
performing archive operations for 70
performing restore operations for 71
recovering files saved by 69
inutoc command 76
inuumsg command 76
Inventory Scout server daemon
launching
using invscoutd command 80
inverted index
building for bibliography 36
invscoutd command 80
ioo command 86
iostat command 96
ip security export files
import conversion 105
ip security packets list 106
ip security tracing buffer 106
ipcrm command 100
ipcs command 101
ipreport command 105
ipsec_convert command 105
ipsecstat command 106
ipsectrcbuf command 106
iptrace daemon 107
lsallqdev command 288
isC2host command 109
isCChost command 110
lsconn command 314
lsdev command 317
lsparent command
listing acceptable connection types from
using lsparent command 350
istat command 111

J

job control 112
jobs command 112
join command 114
joining data fields
from two files
using join command 114
joinvg command 117

K

kdb command 118
key
binding
using mwm command 687
keyadd command 125
keyboard mapping file
into an input method keymap file, compiling
using keycomp command 127
keyboard maps
listing
using lskbd command 330
keycomp command 127
keydelete command 129
keyenvoy command 130
keylist command 130
keylogin command 132
keypasswd command 133
keyserv daemon 135
intermediary between user processes
using keyenvoy command 130

keywords
installing filesets associated with 63
kill command 136
killall command 138
kmodctrl command 142
Korn shell
invoking 147
krlogind daemon
server function
providing 144
krshd daemon
server function
providing 145
ksh command 147

L

last command 151
lastcomm command 152
lastlogin command 154
launching
Inventory Scout server daemon
using invscoutd command 80
lb_admin
add command 155
clean command 155
command description of 154
delete command 156
help command 156
lookup command 156
quit command 156
register command 156
set_broker command 157
set_timeout command 157
unregister command 157
use_broker command 157
lb_find command 157
ld command 160
address maps 176
archive files 161
attributes of exported symbols 175
binder 165
environment variables 177
examples 177
files 178
garbage collection 162
ignored and unsupported flags 163
import and export file format
-bI: and -bE: Flags 173
import and export files 161
libraries 161
linking mode 161
options (-bOptions) 165
processing 162
run-time linking 173
symbols 162
ldd command 178
ldedit 179
learn command 180
leave command 181
lex command
definitions 185
rules 186
specification file 185
lexical analyzer 184
libraries
checking for incompatibilities 194
using ld command 161
line command 191
line printer
generating a ripple pattern for a
using lptest command 277
sending requests to
using lp command 225
line printer spooling queue
removing jobs from
using lprm command 267
link command 192
link subroutine 192
linking mode
using ld command 161
links
hard 202
symbolic 202
lint command 193
listdgrp command 198
listing
software products
using lslpp command 332
listvgbackup command 199
listX11input command 201
llbd (local location broker daemon)
description of 201
ln command 202
locale command
writes information about locales 204
localedef command
Processes locale and character map files 206
location broker 157
administrative tools for 154
lookup of 156
registering, via lb_admin 156
lock command 208
lock requests
processing
using lockd daemon 209
lockd daemon 209
locktrace command
controls kernel lock tracing 211
logevent command 212
logevent script 212
logform command 214
logger command 215
logical volume
adding mirrors to
using mklvcopy command 529
creating
using mklv command 523
displaying information about
using lslv command 336

logical volume *(continued)*
mirroring
using mirrorvg command 470
login command 216
login name
displaying the current process
using logname command 221
logins command 218
logname command 221
logoffs
displaying all previous 151
logout command 222
look command 222
lookbib command 223
lookup, command in lb_admin 156
lorder command 224
lp command 225
lpd command 245
lppchk command 256
lppmgr command 258
lpq command 260
lpr command 263
lprm command 267
lpstat command 271
lptest command 277
ls command 278
ls-secldapclntd 282
lsactdef command 283
lsallq command 287
lsattr command 289
lsaudrec command 294
lsauthent command
authentication methods 298
lsC2admin command 298
lsCCadmin command 299
lscfg command 300
lsclass command 302
lscomg command 303
lscondition command 306
lscondresp command 310
lscons command 316
lsdisp command 324
lsfilt command
listing filter rules 325
lsfont command 326
lsfs command 327
lsgroup command 328
lsitab command 330
lskbd command 330
lslicense command
fixed and floating licenses
listing number and status of 331
lslpp command 332
lslv command 336
lsmaster command 339
lsmcode command 340
lsmksysb command 342
lsnamsv command 344
lsnfsexp command 345
lsnfsmnt command 346
lsnim command 347
lsnlspath command 350
lspath command 352
lsprtsv command 357
lsps command 358
lspv command 359
lsque command 362
lsquedev
command 364
lsquedev command 364
lsresource command
displaying bus resources 365
lsresponse command 368
lsrole command 372
lsrpdomain command 373
lsrpnode command 376
lsrset 378
lsrsrc command 379
lsrsrcdef command 384
lssavevg command 389
lssec
listing attributes of
security stanza files 391
lssensor command 393
lssrc command 397
lstun command
listing tunnel definitions 399
lsuser command 400
lsvfs command 402
lsvg command 403
lsvirprt command 407
lsvmode command 410
displaying current video mode 410
lswlmconf command 411
lvmstat 414

M

m4 command
preprocessing files 416
mach command 419
machstat command 420
macref command
producing cross-reference listing of 420
macro file 420
mail
incorporating into a folder 30
receiving
using mail command 422
using Mail command 422
sending 463
using mail command 422
using Mail command 422
mail command 422
Mail command 422
Mail commands
mailq 434
mailstats 436
mail queue
printing contents of
using mailq command 434

mail queue messages
printing list of
using mailq command 434
mail traffic statistics
displaying 436
mailbox directories, setting up 48
mailq command 434
mailstats command 436
mailx command 422
maintaining groups of programs
using make command 437
make command 437
makedbm command 443
makedepend command 444
makedev command 447
man command 449
man pages
displaying information online 449
managefonts command 452
mant command 454
manual
typesetting pages of
using mant command 454
mark command 455
memorandum macro 607
menu pane
window manager
using mwm command 687
mesg command 458
message facility commands
mkcatdefs 479
message queues
removing identifiers 100
message sequences
creating
using mark command 455
displaying
using mark command 455
modifying
using mark command 455
message source file
preprocessing 479
messages
checking for
using msgchk command 642
permitting
using mesg command 458
printing full path names of
using mhpath command 464
producing formatted lists of
using mhl command 461
refusing
using mesg command 458
mgmtsvr command 459
MH
install_mh command 48
MH shell
creating
using msh command 643
mhl command 461
mhmail command 463
mhpath command 464
microcode and firmware levels of system
displaying
using lsmcode command 340
migratelp 466
migratepv command 467
mirrord command
controlling and monitoring
mirror monitors for maintenance 468
mirrorvg command 470
mkboot command 476
mkC2admin command 478
mkcatdefs command 479
mkCCadmin command 481
mkcd command 482
mkcfsmnt command 486
mkclass command 487
mkclient command 489
mkcomg command 490
mkcondition command 493
mkcondresp command 498
mkdev command 500
mkdir command 503
mkdirhier command 504
mkfifo command
making FIFO special files 505
mkfilt command
activating or deactivating filter rules 506
mkfont command 507
mkfontdir command 507
mkfs command 509
mkgroup command
description of 511
mkhosts command 514
mkinstallp command 517
mkitab command 515
mkkeyserv command 518
mklost+found command 523
mklv command 523
mklvcopy command 529
mkmaster command 531
mknamsv command 532
mknetid command 534
mknfs command 535
mknfsexp command 536
mknfsmnt command 537
mknod command 540
mkpasswd command 542
mkpath command 544
mkproto command 546
mkprtldap command 550
mkprtsv command 553
mkps command 556
mkque command 558
mkquedev command 559
mkramdisk command 561
mkresponse command 563
mkrole command 567
mkrpdomain command 569
mkrset 571
mkrsrc command 572

mksecpki command 581
mksensor command 583
mkslave command 586
mkssys command 587
mkstr command 590
mksysb command 591
mkszfile command 594
mktcpip command 595
mkuser command 597
mkuser.sys command 600
mkvg command 601
mkvgdata command 604
mkvirprt command 605
mm command
printing document formatted with
using mm command 607
mmt command 609
mmtu command 611
mobip6ctrl commnad 611
mobip6reqd daemon 613
mode, linking
using ld command 161
modify XCOFF header 179
monacct command 614
monitord daemon 616
mosy command 622
mount command 624
mountd daemon 633
mounting
automatic
using mount command 625
CacheFS directory
using mkcfsmnt command 486
mounting a directory from
NFS server
using mknfsmnt command 537
mpcfg command 634
mrouted daemon 638
msgchk command 642
msh command 643
mtrace command 646
multicast datagram
forwarding a
using mrouted daemaon 638
multicast path
from a source to a receiver, printing a
using mtrace command 646
MultiPath I/O
lspath command 352
mkpath command 544
mvt command 651
MWM 653
mwm command 653

N

NCS daemons
llbd 201
Network Install Manager 347
new roles
creating
using mkrole command 567
NFS
configuring system to run
using mknfs command 535
NFS commands
mknfs 535
mknfsexp 536
mknfsmnt 537
NFS daemons
lockd 209
mountd 633
NFS server
mounting a directory from
using mknfsmnt command 537
NIM commands
lsnim 347
NIS commands
keyenvoy 130
keylogin 132
lsmaster 339
lsnfsexp 345
lsnfsmnt 346
makedbm 443
mkclient 489
mkkeyserv 518
mkmaster 531
mkslave 586
NIS daemons
keyserv 135
NLSPATH,
secure system configuration variable,
showing the value of 350
notify object class (SRC)
adding a notify method definition 541

O

object files
linking
using ld command 160
reading one or more 224
objects
shared
using ld command 161
options (-bOptions)
in ld command 165

P

packet tracing
ipreport command 105
iptrace daemon 107
paging space
adding an additional
using mkps command 556
displaying the characteristics of
using lsps command 358

password
reserving a terminal
using lock command 208
performing simply copy operations 65
physical volume
moving partitions from one volume to another volume
using migratepv command 467
plotter queue
listing device names with a queue
using lsallqdev command 288
policy
keyboard input
using mwm command 656
port
stopping all processes on a
using logout command 222
port 515
monitoring for print requests
using lpd command 245
print server
providing on a network
using lpd command 245
print services
configuring TCP/IP-based
configuring 553
printer
enqueuing
using lpr command 263
line
sending requests to a 225
printer queue
adding to the system
using mkque command 558
using mkquedev command 559
listing device names with a queue
using lsallqdev command 288
printing a multicast path
from a source to a receiver
using mtrace command 646
process
stopping 136
process groups 112
processes
canceling all 138
stopping all port
using logout command 222
program products
listing optional
using lslpp command 332
programs
creating make file dependencies 444
installing compatible
using installp command 50
providing a C preprocessor interface to the make command 23
programs, groups of
maintaining
using make command 437
regenerating
using make command 437
programs, groups of *(continued)*
updating
using make command 437

Q

queue
displaying the stanza name for the
using lsquedev command 364
displaying the stanza name of
using lsque command 362
queues
listing the names of all configured
using lsallq command 287
quit, command in lb_admin 156

R

RAM disk 561
read operations
copying from standard input 191
receiving mail
using mail command 422
records, makes
in /etc/inittab file
using mkitab command 515
regenerating groups of programs
using make command 437
register, command in lb_admin 156
resource
client-specific
using mwm command 674
resource description file
using mwm command 679
resource set
Client-Specific 675
Component Appearance 659
Frame and Icon Component 661
General Appearance and Behavior 662
roles
creating new
using mkrole command 567
run-time linking
using ld command 173

S

screens
displaying text one screen at a time 618
scripts
elogevent 212
event response resource manager (ERRM)
elogevent 212
logevent 212
logevent 212
sending mail
using mail command 422
service record
creating
making mkboot command 476

session
initiating a user
using login command 216
session management
stopping client
using mwm command 653
set_broker, command in lb_admin 156
set_timeout, command in lb_admin 157
shared objects
using ld command 161
shells
Korn 147
standard 147
slide
typesetting
using mvt command 651
smit
turning on/off maintenance of I/O history 97
spool queue
examining
using lpq command 260
spooling queue
removing jobs from the line printer 267
SRC
adding a definition to subsystem object class
mkssys command examples 589
using mkssys command 587
adding a subserver object 585
mkserver command example 585
adding a subsystem notification method 541
mknotify command examples 542
SRC configuration commands
mknotify 541
mkserver 585
mkssys 587
standard input
reading 191
standard output
writing 191
stanza name, device
displaying the
using lsquedev command 364
storing public and private keys 135
subserver
getting status
using lssrc command 397
subservers
adding a definition to SRC object class 585
mkserver command example 585
subsystem
getting status
using lssrc command 397
subsystems
adding a definition to SRC object class
mkssys command examples 589
using mkssys command 587
adding a notification method 541
mknotify command examples 542
system
adding a device to
using mkdev command 500
system images, displaying
for examining a dump
using kdb command 118
system log
making entries in
using the logger command 215
system resource controller 541, 585, 587

T

tape device
giving subcommands
mt command 644
TCP/IP
hosts
generating table file 514
setting required values 595
internet packets 107
name service
displaying information 344
name services
configuring 532
network interface
interface, configuring 8
network options
displaying, adding, and deleting 611
packet trace report
generating 105
print services
configuring 553
displaying information about 357
service management 37
TCP/IP commands
ifconfig 8
ipreport 105
lsnamsv 344
lsprtsv 357
mkhosts 514
mknamsv 532
mkprtsv 553
mktcpip 595
mmtu 611
TCP/IP daemons
inetd 37
iptrace 107
TCP/IP smit commands
lsnamsv 344
lsprtsv 357
mknamsv 532
mkprtsv 553
mktcpip 595
terminal
reserving 208
time management
creating reminders 181
timeout setting in lb_admin 157
tty
reporting statistics 96
typesetting document
using mmt command 609

U

unregister, command in lb_admin 157
updating
  instructions
    getting copies of 66
  procedure
    saving files changed during 73
updating groups of programs
  using make command 437
use_broker
  command in lb_admin 157
user
  initiating a session for a
    using login command 216
user account
  creating
    using mkuser command 597
  customizing
    using mkuser.sys command 600
user database
  creating a version for searches
    using mkpasswd command 542
users
  displaying
    system ID 6
  displaying attributes
    using lsuser command 400
  displaying role attributes
    using lsrole command 372

V

verify files
  installable software product
    using lppchk command 256
view graph
  typesetting
    using mvt command 651
virtual printer
  displaying the attribute values of
    using lsvirprt command 407
  making
    using mkvirprt 605
virtual RAM disk
  making 561
volume group
  creating
    using mkvg command 601
  displaying information about
    using lsvg command 403
  displaying information about a physical volume
    using lspv command 359
  importing new definition 27

W

window
  default components 654
  default menu
    using mwm command 654
window *(continued)*
  iconifying
    using mwm command 655
  menu pane
    using mwm command 688
window manager
  event specification
    using mwm command 685
window stacking 657
Workload Manager (WLM)
  listing configurations 411
  lswlmconf command 411

X

x defaults 657
X Window System
  window management
    using mwm command 653
XCOFF
  modify 179

Y

ypbind daemon
  starting
    using mkclient command 489

Vos remarques sur ce document / Technical publication remark form

Titre / Title : Bull AIX 5L Commands Reference Volume 3/6

Nº Reférence / Reference Nº : 86 A2 42EF 02

Daté / Dated : May 2003

ERREURS DETECTEES / ERRORS IN PUBLICATION

AMELIORATIONS SUGGEREES / SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Vos remarques et suggestions seront examinées attentivement.
Si vous désirez une réponse écrite, veuillez indiquer ci-après votre adresse postale complète.

Your comments will be promptly investigated by qualified technical personnel and action will be taken as required.
If you require a written reply, please furnish your complete mailing address below.

NOM / NAME : _____ Date : _____

SOCIETE / COMPANY : _____

ADRESSE / ADDRESS : _____

_____

Remettez cet imprimé à un responsable BULL ou envoyez-le directement à :

Please give this technical publication remark form to your BULL representative or mail to:

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

Technical Publications Ordering Form

Bon de Commande de Documents Techniques

To order additional publications, please fill up a copy of this form and send it via mail to:
Pour commander des documents techniques, remplissez une copie de ce formulaire et envoyez-la à :

BULL CEDOC
ATTN / Mr. L. CHERUBIN
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

Phone / Téléphone : +33 (0) 2 41 73 63 96
FAX / Télécopie +33 (0) 2 41 73 60 19
E–Mail / Courrier Electronique : srv.Cedoc@franp.bull.fr

Or visit our web sites at: / Ou visitez nos sites web à:

http://www.logistics.bull.net/cedoc
http://www-frec.bull.com **http://www.bull.com**

CEDOC Reference # Nº Référence CEDOC	**Qty** Qté	**CEDOC Reference #** Nº Référence CEDOC	**Qty** Qté	**CEDOC Reference #** Nº Référence CEDOC	**Qty** Qté
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
__ __ _____ _ [__]		__ __ _____ _ [__]		__ __ _____ _ [__]	
[__] : **no revision number means latest revision** / pas de numéro de révision signifie révision la plus récente					

NOM / NAME : _____ Date : _____

SOCIETE / COMPANY : _____

ADRESSE / ADDRESS : _____

_____

PHONE / TELEPHONE : _____ FAX : _____

E–MAIL : _____

For Bull Subsidiaries / Pour les Filiales Bull :

Identification: _____

For Bull Affiliated Customers / Pour les Clients Affiliés Bull :

Customer Code / Code Client : _____

For Bull Internal Customers / Pour les Clients Internes Bull :

Budgetary Section / Section Budgétaire : _____

For Others / Pour les Autres :

Please ask your Bull representative. / Merci de demander à votre contact Bull.

PLACE BAR CODE IN LOWER LEFT CORNER

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

ORDER REFERENCE
86 A2 42EF 02

Bull

Utiliser les marques de découpe pour obtenir les étiquettes.
Use the cut marks to get the labels.

AIX
AIX 5L Commands Reference

Volume 3/6
86 A2 42EF 02

AIX
AIX 5L Commands Reference

Volume 3/6
86 A2 42EF 02

AIX
AIX 5L Commands Reference

Volume 3/6
86 A2 42EF 02