AIX Version 7.2

Commands Reference, Volume 3, i- m

AIX Version 7.2

Commands Reference, Volume 3, i- m

Note
Before using this information and the product it supports, read the information in "Notices" on page 965.

This edition applies to AIX Version 7.2 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2015, 2017.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this document vii
Highlighting vii
Case sensitivity in AIX vii
ISO 9000 vii
Support for the single UNIX specification vii

i 1
ibm3812 Command 1
ibm3816 Command 2
ibm5585H-T Command 3
ibm5587G Command 4
ibstat Command 5
iconv Command 7
id Command 8
ifconfig Command 11
ike Command 20
ikedb Command. 24
imake Command 27
imapd Daemon 30
imapds Daemon. 31
impfilt Command 32
importvg Command 32
imptun Command 35
inc Command 36
indent Command 38
indxbib Command 42
inetd Daemon 43
infocmp Command. 45
telinit or init Command 48
install Command 52
install_all_updates Command 54
install_assist Command 57
install_mh Command 58
installbsd Command 59
installios Command 60
installp Command 62
instfix Command 73
inucp Command. 75
inudocm Command 77
inulag Command 78
inurecv Command 80
inurest Command 81
inurid Command 83
inusave Command 84
inutoc Command 87
inuumsg Command 88
inuwpar Command. 89
invscout Command. 90
invscoutd Command 95
ioo Command 100
iostat Command 111
ipcrm Command 121
ipcs Command 122
ipfilter Command 125
ipreport Command 126
ipsec_convert Command 127
ipsecstat Command 127
ipsectrcbuf Command 128
iptrace Daemon 129
ipv6policy Command. 132
isC2host Command 133
isCChost Command 134
isnstgtd Command 135
istat Command. 136

j 139
j2edlimit Command 139
jobs Command 141
join Command 143
joinvg Command 146

k 149
kdb Command 149
kdestroy Command 151
keyadd Command. 152
keycomp Command 154
keydelete Command 155
keyenvoy Command 157
keylist Command 157
keylogin Command 159
keypasswd Command 160
keyserv Daemon 161
kill Command 162
killall Command 164
kinit Command 165
klist Command. 167
kmodctrl Command 169
kpasswd Command 169
krlogind Daemon 170
krshd Daemon 172
ksh Command 173
ksh93 Command 176
kvno Command 179

l 181
labcat Command 181
labck Command 182
last Command 184
lastcomm Command 186
lastlogin Command 187
lbxproxy Command 188
ld Command 190
ldd Command 211
ldedit Command 212
learn Command 214
leave Command 216
lecstat Command 216
lex Command 218
line Command 224
link Command 225

© Copyright IBM Corp. 2015, 2017

lint Command 226
listdgrp Command 231
listvgbackup Command 231
listX11input Command 233
livedumpstart Command 234
lkdev Command 238
ln Command 239
locale Command 241
localedef Command 243
lock Command 245
lockd Daemon 245
locktrace Command 247
logevent Command 248
logform Command 250
logger Command 252
login Command 253
logins Command 256
logname Command 259
logout Command 260
look Command 261
lookbib Command 261
loopmount Command 262
loopumount Command 264
lorder Command 265
lp Command 266
lp.cat, lp.set, lp.tell Command 272
lpacl Information 274
lpadmin Command 277
lpar_netboot Command 285
lparstat Command 288
lpc Command 297
lpd Command 299
lpfilter Command 301
lpforms Command 306
lphistory Command 310
lpmove Command 314
lppchk Command 315
lppmgr Command 317
lpq Command 319
lpr Command 323
lprm Command 327
lpsched Command 330
lpstat Command 331
lpsystem Command 335
lptest Command 337
lpusers Command 338
ls Command 339
ls-secldapclntd Command 344
lsactdef Command 345
lsallq Command 349
lsallqdev Command 350
lsarm command 351
lsassocmap Command 352
lsattr Command 354
lsaudrec Command 359
lsauth Command 363
lsauthent Command 365
lsC2admin Command 366
lsCCadmin Command 367
lscfg Command 367
lscifscred Command 370
lscifsmnt Command 371
lsclass Command 372
lscluster Command 374
lscomg Command 377
lscondition Command 380
lscondresp Command 385
lsconn Command 390
lscons Command 391
lscore Command 393
lscosi Command 394
lsdev Command 396
lsdisp Command 403
lsdom Command 403
lsevent Command 405
lsfilt Command 409
lsfont Command 410
lsfs Command 411
lsgroup Command 412
lsiscsi Command 414
lsitab Command 416
lskbd Command 417
lskst Command 417
lsldap Command 419
lslicense Command 423
lslpclacl Command 424
lslpcmd Command 429
lslpp Command 432
lslpracl Command 437
lslpriacl Command 442
lslprsacl Command 447
lslv Command 452
lsmaster Command 455
lsmcode Command 456
lsmksysb Command 458
lsmp Command 460
lsmpio Command 461
lsnamsv Command 466
lsnfsexp Command 467
lsnfsmnt Command 468
lsnim Command 469
lsnlspath Command 472
lsparent Command 472
lspath Command 474
lspriv Command 479
lsprtsv Command 480
lsps Command 480
lspv Command 482
lspprc Command 485
lsque Command 487
lsquedev Command 488
lsresource Command 490
lsresponse Command 493
lsrole Command 497
lsrpdomain Command 499
lsrpnode Command 502
lsrset Command 506
lsrsrc Command 507
lsrsrcassoc Command 513
lsrsrcdef Command 516
lssavevg Command 522
lssavewpar Command 524

lssec Command 525
lssecattr Command 528
lssecmode Command 531
lssensor Command 532
lsslot Command 539
lssrad Command 541
lssrc Command 542
lsts Command 545
lstun Command 547
lstxattr Command 548
lsuser Command 550
lsusil Command 553
lsvfs Command 553
lsvg Command 554
lsvgfs Command 557
lsvirprt Command 558
lsvmode Command 561
lsvpd Command 562
lsvsd Command 567
lswlmconf Command 569
lswpar Command 573
luit Command 584
lvmo Command 585
lvmstat Command 588
lvupdateInit Command 591
lvupdateRegKE Command 592
lvupdateRegScript Command 593
lvupdateSafeKE Command 595
lvupdateSetProcs Command 596

m 599
m4 Command 599
mach Command 602
machstat Command 602
macref Command 603
mail, Mail, or mailx Command 605
mailq Command 616
mailstats Command 617
make Command 618
makedbm Command 625
makedepend Command 626
makedev Command 629
makekey Command 629
makemap Command 630
man Command 631
manage_disk_drivers Command 636
managefonts Command 637
mant Command 639
mark Command 640
mesg Command 643
mhl Command 644
mhmail Command 646
mhpath Command 647
migratelp Command 649
migratepv Command 650
migwpar Command 652
mirrorvg Command 653
mirscan Command 656
mkauth Command 659
mkboot Command 661
mkC2admin Command 663
mkcatdefs Command 664
mkCCadmin Command 666
mkcd Command 667
mkcfsmnt Command 672
mkcifscred Command 673
mkcifsmnt Command 674
mkcimreg Command 676
mkclass Command 679
mkclient Command 682
mkcluster Command 683
mkcomg Command 686
mkcondition Command 690
mkcondresp Command 697
mkcosi Command 699
mkdev Command 701
mkdir Command 703
mkdirhier Command 705
mkdom Command 705
mkdvd Command 707
mkfifo Command 712
mkfilt Command 713
mkfont Command 714
mkfontdir Command 714
mkfs Command 716
mkgroup Command 719
mkhosts Command 723
mkiba Command 724
mkinstallp Command 725
mkiscsi Command 727
mkitab Command 728
mkkeyserv Command 730
mkkrb5clnt Command 731
mkkrb5srv Command 733
mklost+found Command 735
mklpcmd Command 736
mklv Command 739
mklvcopy Command 745
mkmaster Command 748
mknamsv Command 749
mknetid Command 750
mknfs Command 751
mknfsexp Command 752
mknfsmnt Command 755
mknfsproxy Command 758
mknod Command 759
mknotify Command 761
mkpasswd Command 762
mkpath Command 764
mkprojldap Command 766
mkproto Command 768
mkprtldap Command 772
mkprtsv Command 775
mkps Command 778
mkqos Command 780
mkque Command 780
mkquedev Command 782
mkramdisk Command 783
mkresponse Command 785
mkrole Command 790
mkrpdomain Command 792
mkrset Command 801

mkrsrc Command 802
mkrtc Command 806
mkseckrb5 Command 808
mksecldap Command 809
mksecpki Command 815
mksensor Command 817
mkserver Command 823
mkslave Command 824
mkssys Command 825
mkstr Command 828
mksysb Command 829
mkszfile Command 833
mktcpip Command 835
mkts Command 837
mktun Command 839
mkuser Command 839
mkuser.sys Command 843
mkusil Command 844
mkvg Command 845
mkvgdata Command 850
mkvirprt Command 851
mkwpar Command 853
mkwpardata Command 866
mm Command 867
mmt Command 869
mmtu Command 871
mobip6ctrl Command 872
mobip6reqd Daemon 873
monacct Command 874
mon-cxma Command 875
monitord Daemon 877
moo Command 878
more Command 878
mosy Command 883
mount Command 885
mountd Daemon 896
mpcstat Command 897
mpio_get_config Command 899
mpstat Command 901
mrouted Daemon 906
msgchk Command 909
msh Command 910
mt Command (BSD) 912
mtrace Command 913
multibos Command 915
mv Command 919
mvdir Command 922
mvfilt Command 923
mvt Command 924
mwm Command 925

Notices 965
Privacy policy considerations 967
Trademarks 967

Index 969

About this document

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

Highlighting

The following highlighting conventions are used in this document:

Bold	Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Bold highlighting also identifies graphical objects, such as buttons, labels, and icons that the you select.
Italics	Identifies parameters for actual names or values that you supply.
`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 text that you must type.

Case sensitivity in AIX

Everything in the AIX operating system is case sensitive, which means that it distinguishes between uppercase and lowercase letters. For example, you can use the **ls** command to list files. If you type `LS`, the system responds that the command `is not found`. Likewise, **FILEA**, **FiLea**, and **filea** are three distinct file names, even if they reside in the same directory. To avoid causing undesirable actions to be performed, always ensure that you use the correct case.

ISO 9000

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

Support for the single UNIX specification

The AIX operating system is designed to support The Open Group's Single UNIX Specification Version 3 (UNIX 03) for portability of operating systems based on the UNIX operating system. Many new interfaces, and some current ones, have been added or enhanced to meet this specification. To determine the correct way to develop a UNIX 03 portable application, see The Open Group's UNIX 03 specification on The UNIX System website (http://www.unix.org).

© Copyright IBM Corp. 2015, 2017

i

The following AIX commands begin with the letter *i*.

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.

© Copyright IBM Corp. 2015, 2017

Flags

Item	Description
-altpaper	Specifies that the file should be printed from the alternate paper drawer. By default, the **ibm3812** 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 **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

Item	Description
/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 reference:

"ibm3816 Command"

Related information:

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

Item	Description
-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

Item	Description
/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 reference:

"ibm3812 Command" on page 1

Related information:

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

Item	Description
-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

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

Related information:

troff command

---

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

Item	Description
-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

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

Related information:

troff command

---

ibstat Command

Purpose

Displays operational information about one or more InfiniBand network devices.

Syntax

ibstat [**-d**, **-h**, **-i**, **-n**, **-p**, **-v**] [*DeviceName*]

Description

This command displays InfiniBand operational information pertaining to a specified Host Channel Adapter Device (HCAD). If an HCAD device name is not entered, status for all available HCADs are displayed. Select a flag to narrow down your search results. You can display specific categories of information, including Node, Port, Interface, and Debug information. You can also choose to display all of the information categories.

Flags

Item	Description
-d	Displays current debug setting.
-h	Displays **ibstat** command usage.
-i	Displays network interface information.
-n	Displays only IB node information.
-p	Displays only IB port information.
-v	Displays all IB device information.

The following fields display information for all valid calls:

Device Name
Displays the name of an available HCAD (for example, `iba0`).

Port State
Displays the current state of each HCAD port.

Down Port is disabled.

Initialized
Port is enabled and issuing training sequences.

Armed
Port is trained and attempting to configure to the active state.

Active Port is in a normal operational state.

Unknown
Port is in an invalid or unknown state.

Parameters

Item	Description
DeviceName	Specifies the name of the HCAD device (for example, `iba0`) **Tip:** The device name is optional. If you do not specify a device name, all InfiniBand devices are queried for control or information.

Exit Status

When you specify an invalid *DeviceName*, the **ibstat** command produces error messages stating that it could not connect to the device. For example:

```
IBSTAT: No device iba2 configured.
```

or:

```
IBSTAT: Device iba3 is not available.
```

Examples

1. To request node and port information, enter:

   ```
   ibstat -n -p
   ```

   Information similar to the following is displayed:

   ```
   ============================================================================
    INFINIBAND DEVICE INFORMATION (iba0)
   ============================================================================

   ----------------------------------------------------------------------------
    IB NODE INFORMATION (iba0)
   ----------------------------------------------------------------------------
   Number of Ports:                         2
   Globally Unique ID (GUID):               00.02.55.00.00.00.46.00
   Maximum Number of Queue Pairs:           1023
   Maximum Outstanding Work Requests:       32768
   Maximum Scatter Gather per WQE:          252
   Maximum Number of Completion Queues:     1023
   Maximum Multicast Groups:                256
   Maximum Memory Regions:                  3836
   Maximum Memory Windows:                  3836

   ----------------------------------------------------------------------------
    IB PORT 1 INFORMATION (iba0)
   ----------------------------------------------------------------------------
   ```

```
Global ID Prefix:                       fe.80.00.00.00.00.00.00
Local ID (LID):                         0012
Port State:                             Active
Maximum Transmission Unit Capacity:     2048
Current Number of Partition Keys:       1
Partition Key List:
  P_Key[0]:                             ffff
Current Number of GUID's:               1
Globally Unique ID List:
  GUID[0]:                              00.02.55.00.00.00.46.12

------------------------------------------------------------------------------
 IB PORT 2 INFORMATION (iba0)
------------------------------------------------------------------------------
Global ID Prefix:                       fe.80.00.00.00.00.00.00
Local ID (LID):                         0011
Port State:                             Active
Maximum Transmission Unit Capacity:     2048
Current Number of Partition Keys:       1
Partition Key List:
  P_Key[0]:                             ffff
Current Number of GUID's:               1
Globally Unique ID List:
  GUID[0]:                              00.02.55.00.00.00.46.52
```

Location

/usr/sbin/ibstat

Related information:
Internet Protocol over InfiniBand

---

iconv Command

Purpose

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

Syntax

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

iconv -l

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 the System Management Interface Tool (SMIT) **smit iconv** fast path to run this command. The **iconv** command uses the **LOCPATH** environment variable to search for code-set converters of the form **iconv**/*FromCodeSet_ToCodeSet*. The default value of **LOCPATH** is **/usr/lib/nls/loc**.

Flags

Item	Description
-c	Omits characters that cannot be converted in the input file from the output. Characters that cannot be converted include characters that are invalid in the *FromCode* of the input or that have no corresponding character in the *ToCode* of the output. After omitting an unconvertible character, **iconv** advances to the next byte of the input to convert the next character. If **-c** is not used, **iconv** exits upon encountering a character that cannot be converted in the input. The presence or absence of **-c** does not affect the exit status of **iconv**.
-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.
-l	Writes all supported *FromCode* and *ToCode* values to standard output.
-s	Suppresses any messages written to standard error concerning invalid characters. When -s is not used, an error message is written to standard error for each unconvertible or truncated character. The presence or absence of **-s** does not affect the exit status of **iconv**.
-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.

Exit Status

This command returns the following exit values:

Item	Description
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:

genxlt command

iconv_open command

Converters Overview

List of Converters

---

id Command

Purpose

Displays the system identifications of a specified user.

Syntax

id [user]

id - G [-n] [*User* **]**

id -g [-n 1 | [-n -r] [*User* **]**

id -u [**-n l** | [**-n r**] [*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, when specified with the **-l** option, displays login UID. Login ID indicates the system credentials at the time of logging in to the session. Login UID indicates the user ID (numeric value) of the user, who actually logged in. The login UID is equal to the UID for a user who has logged in to the system and whose credentials remain unchanged. For example, when the user runs the **su** command, the UID for the user changes and the login UID remains the same.

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:

Item	Description
-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.
-l	Specifies that the **id** command write the login ID instead of the real or effective ID. This flag can be invoked with either the **-u** flag to write the login UID or the **-g** flag to write the primary group ID for the login user. When *username* is passed with the **-l** option, the **id** command displays the ID details of the user name instead of the login ID details.
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:

Item	Description
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
   ```
5. To display the login UID after logging in as root and running the **su** command to user **sah**, type:
   ```
   id -lu
   ```

   Output is displayed in the following format:
   ```
   0
   ```
6. To display the primary group name of the user who actually logged in, type:
   ```
   id -lgn
   ```

   Output is displayed in the following format:
   ```
   system
   ```
7. To display the primary group ID of the user who actually logged in, type:
   ```
   id -lg
   ```

Output is displayed in the following format:

```
0
```

Files

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

Related reference:

"login Command" on page 253

Related information:

getty command

setgroups command

su command

ifconfig Command

Purpose

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

Syntax

ifconfig *interface* [*addressfamily* [*address* [*destinationaddress*]] [*parameters...*]]

ifconfig *interface* [*protocolfamily*] *interface protocolfamily*

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

ifconfig *interface* [**tcp_low_rto** *rto* | **-tcp_low_rto**]

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 system. 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 might require separate naming schemes. It is necessary to specify the *addressfamily* parameter, which might change the interpretation of the remaining parameters. The address families that are currently supported are **inet** and **inet6**.

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 that is expressed in the Internet standard dotted decimal notation.

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** reports only the details specific to that protocol family.

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

Gratuitous ARP is supported for Ethernet, token-ring, and FDDI interfaces. This support means when an IP address is assigned, the host sends an ARP request for its own address (the new address) to inform other systems 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 by using **errpt** command (or by using SMIT interface) for the error ID **AIXIF_ARP_DUP_ADDR**.

The **ifconfig** command calls the **ifconfig.ib** command. The **ifconfig.ib** command displays the interface information as shown:

```
ib0:flags=e3a0063<UP,BROADCAST,NOTRAILERS,RUNNING,ALLCAST,MULTICAST,GROUPRT>
pmtu 2048 qkey
0x1e qpn 0x48 lid 0x5c hca iba0 port 1 inet 1.2.3.120 netmask 0xffffff00 broadcast
1.2.3.255 inet6
fe80::2:c903:1:1e8d/64 tcp_sendspace 262144 tcp_recvspace 262144 rfc1323 1
```

The interface now displays the path mtu (pmtu), Queue Key (qkey), Queue Pair Number (qpn), Local ID (lid), Host Channel Adapter (hca), and the port number (port). This information is useful during debugging or performing diagnostics.

Note: Any changes that are made to the attributes of an interface by using the **ifconfig** command are lost when the system is rebooted.

When the **detach** command is specified with the **ifconfig** command, all other options are ignored. Specifying the **detach** command prevents the garbage character in the **ipconfig** command from causing any error. For example, the `ifconfig en3 garbage detach` command runs properly even when with the bad option.

Flags

Item	Description
-a	Optionally, the **-a** flag can 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. You can use the flag only with the **-a** or **-l** flag.
-l	This flag can 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. You can use the flag only with the **-a** or **-l** flag.
protocolfamily	This flag specifies protocols such as **tcp**, **udp**, **tcp6**, **udp6**, **icmp**, and **icmp6**.

Parameters

Item	Description
address	Specifies the network address for the network interface. For the **inet** family, the *address* parameter is either a host name or an IP address in the standard dotted decimal notation.
addressfamily	Specifies which network address family to change. The **inet** and **inet6** 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.

Item	Description
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 asynchronous transfer mode (ATM) • en for Standard Ethernet (inet) • et for IEEE 802.3 Ethernet (inet) • gre for generic routing encapsulation tunnel pseudointerface (inet) • gif for IPv4-over-IPv6 tunnel pseudointerface (inet) • tr for token-ring (inet) • 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) • ib for IP over InfiniBand (inet) • tap for TAP pseudo-Ethernet 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 that are 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. **anycast** (inet6 only) Adds the specified anycast address. **-anycast** (inet6 only) Deletes the specified anycast address. **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 1's. **checksum_offload** Enables the flag to indicate that the transmit TCP checksum will be offloaded to the adapter. The command also resets the per-interface counter that determines whether TCP must dynamically enable or disable offloading of checksum computation. **-checksum_offload** Disables transmit TCP checksum offloading. **create** (TAP only) Creates a network interface. You can either create a specific interface, such as **tap0** or specify the **tap** option to create the next available TAP interface, such as **ifconfig tap create**.

Item	Description
-dad	(inet6 only) Does not perform duplicate IPv6 address detection.
-debug	Disables driver-dependent debug code.
delete	Removes the specified network address. This command 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 lets you specify the host portion again.
destroy	(TAP only) Destroys a network interface. The **ifconfig** command removes the specified network interface from the list of interfaces. It also removes the interface and any associated TAP network device from the Object Data Manager (ODM).
device *dev_name*	(ATM network interface only). Specifies the device name that this interface is associated with. Unlike token ring or Ethernet, in case of ATM, there is not a one-to-one correspondence between an interface and a 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. 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 the reception of messages. Routes that use the interface, however, are not automatically disabled.
eui64	(inet6 only) Computes real IPv6 address by replacing the last 64 bytes of the specified address with the Interface Identifier.
first	Puts an IPv6 address at the first place on an interface to select it as the source for unbound sockets. The syntax for using this parameter is as follows: `ifconfig interface inet6 first address`
firstalias	Same as alias, but sets the address in front of the interface address list 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 that are handled by an interface are sent out by using the associated adapter.
-hwloop	Disables hardware loop-back. The hardware loop-back specifies that locally addressed packets that are handled by an interface must be sent out using the associated adapter.
ipdst	Specifies an Internet host that can receive IP packets encapsulating **ns** packets that are 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.

Item	Description
ipv6dst	Used to specify an IPv6 node that can receive IPv6 packets encapsulating IPv6 or IPv4 packets through a tunnel. The apparent destination of the point-to-point tunnel interface might not be the real destination of the packets. At the tunnel endpoint, the decapsulated packets might then be forwarded to their final destination.
largesend	Enables one LPAR to send large data in a single packet to another LPAR. It works similarly to **largesend** over real adapters except in this case no TCP segmentation is done. If the SEA on VIOS supports **largesend**, the LPAR can transmit large data, which gets segmented by the real adapter on SEA. Use the **chdev** command to enable the **largesend** attribute on SEA.
-largesend	Disables **largesend** over virtual Ethernet. This value is the default.
link [0-2]	Enables special processing of the link level of the interface. These 3 options are interface-specific. In actual effect, however, they are used to select special modes of operation. An example of the usage is to enable Serial Line Internet Protocol (SLIP) compression, or to select the connector type for some Ethernet cards. For more information, see the manual page of the specific driver.
-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. 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 additional hops to the destination network or host.
monitor	Enables the underlying adapter to notify the interface layer of link status changes. The adapter must support link status callback notification. If multipath routing is used, alternative routes are selected when a link goes down.
-monitor	Disables monitoring of the adapter link status.
mtu *Value*	Sets the maximum IP packet size for this system. The *Value* variable can be a number in the range 60 - 65535, but is media-dependent. See Automatic configuration of network interfaces in *Networks and communication management* for the maximum transmission unit (MTU) values by interface.
netmask *Mask*	Specifies how much of the address must be reserved for subdividing networks into subnetworks. This parameter can be used only with the **inet** address family. The *Mask* variable includes both the network part and the subnet part of the local address, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number that begins with 0x, in standard Internet dotted decimal notation, or begins with a name or alias that is listed in the **/etc/networks** file. In the 32-bit address, the *Mask* variable contains 1s for the bit positions that are reserved for the network and subnet parts, and *Mask* variable contains 0s for the bit positions that specify the host. The *Mask* variable contains at least the standard network portion. The subnet segment is contiguous with the network segment. If the **netmask** parameter is used without specifying the IP address, the netmask value of the first IP address of the specified interface is updated.
pvc	(ATM network interface only). Specifies that this interface supports Permanent Virtual Circuit (PVC) types of virtual connections only.

Item	Description
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*	(ATM Network interface only). Specifies that this interface supports both switched virtual circuit (SVC) and PVC types of virtual connections. It further specifies that this interface is an ARP client. The *server_addr* is the list of 20-byte ATM addresses of the ARP servers that this client uses. 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 be separated by a comma.
site6	Sets the IPv6 site number (default is 0). This command must be used only with site-local addresses on a multiple-site node.
svc_s	(ATM network interface only). Specifies that this interface supports both SVC and PVC types of virtual connections. It further specifies that this interface is 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_low_rto	Enables the use of lower retransmission timeouts (RTO) for TCP connections on a low latency, fast network, such as gigabit ethernet and 10-gigabit ethernet). If the networks experience packet drops, the respective TCP connections use the *rto* value for RTO. The *rto* values are in the range of 0 - 3000 ms. This runtime option must be set in the **if_isno** flags field. The **use_isno** option must also be set for this flag to be effective.
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 that is 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 value is the default.
thread	(**inet** only) Configures dedicated kernel threads for an interface. This parameter can be used only SMP systems that have multiple CPU. 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 CPUs running the kernel threads (improved pipelining). For some workloads, this parameter increases the per packet load, due to the thread scheduling load, resulting in higher CPU utilization, and possibly lower throughput.
-thread	(**inet** only) Disables kernel thread support that has been configured with the *thread* parameter.

Item	Description
	tunnel Configures a dedicated tunnel for the trusted communication. A tunnel establishes a virtual link between two trusted nodes for transmitting data packets as payloads of other packet headers. A tunnel can be one of the following types: **Generic routing encapsulation (GRE) tunnel** Expects the source and destination IPv4 addresses of the tunnel endpoint as arguments that are followed by the **tunnel** parameter value. A tunnel is created between the 2 endpoints. **IPv4 over IPv6 tunnel (GIF tunnel)** Expects the source IPv6 address of the tunnel. The address is followed by a destination IPv4 address and a destination IPv6 address that are separated by a comma. For one-to-many tunnels, each target is separated by a comma.
	transfer *tointerface* Transfers an address and its related static routes from *interface* to *tointerface*. For IPv6, this command works only for addresses added by using the **ifconfig** command. `ifconfig interface addressfamily address transfer tointerface` **Note:** If you want to transfer an IP address from one interface to another, and if the destination interface is not part of the virtual LAN (VLAN) to which the IP address belongs, you must add the VLAN to the adapter on which the destination interface is configured.
	up Marks an interface as active (**up**). This parameter is used automatically when you set the first address for an interface. It can also be used to enable an interface after you issue an **ifconfig down** command.
	vipa_iflist Adds the interfaces to the list of interfaces that must use this **vipa** parameter 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.
	scope addrscope zone zoneid Moves the interface into the topological zone that is specified by *zoneid* at the address scope that is specified by the *addrscope*. IPv6 zones are defined in RFC 4007. The parameter is applicable to only inet6.
rto	Specifies the retransmission timeout in milliseconds. The range for this value is 0 - 3000. **Requirement:** You must set the **timer_wheel_tick** value of the **no** command before you set the *rto* value by using the **ifconfig** command. The *rto* value that you specify must be equal to or a multiple of 10 times the **timer_wheel_tick** value that is set.

The following network options, commonly known as Interface Specific Network Options (ISNO), 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 by using this interface will attempt to negotiate the RFC enhancements. A value of 0 disables **rfc1323** for all connections by 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 by using the **setsockopt** subroutine.

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

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

tcp_recvspace *Size*
Specifies the default socket buffer size for interface sockets that are receiving data. The buffer size affects the window size that is used by TCP. (For more information, see the **no** command.)

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

tcp_sendspace *Size*
Specifies the default socket buffer size for interface sockets that are sending data. The buffer size affects the window size that is used by TCP. (For more information, see the **no** command.)

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

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

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

Tip: Parameters that you set by using the **ifconfig** command are lost the next time that you restart your system. Use the **chdev** command to change the Object Data Manager (ODM) database for each interface to make parameter changes permanent. Use the **lsattr -E -l** [*interface*] command to view the interface attributes and use the **chdev -l** [*interface*] **-a** [*attribute=value*] command to change the attribute. For example:

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

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

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

   ```
   ifconfig  sl1
   ```

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

   ```
   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 following command:

   ```
   ifconfig  lo0  inet  127.0.0.1  up
   ```

3. To mark the local token-ring interface as down, enter the following command:

   ```
   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 the following command:
   ```
   ifconfig en0 rfc1323 0
   ```
5. To configure a list of interfaces to use a vipa, enter the following command:
   ```
   ifconfig vi0 vipa_iflist en0,en1,tr0
   ```
6. To remove interfaces that are configured to use vipa, enter the following command:
   ```
   ifconfig vi0 -vipa_iflist en1,tr0
   ```
7. To find out which interfaces are configured to use a vipa, say vi0, enter the following command:
   ```
   ifconfig vi0
   ```
8. To enable link status monitoring, enter the following command:
   ```
   ifconfig en0 monitor
   ```

   If the link status on adapter `ent0` changes to `down`, the adapter notifies the interface layer, which causes the interface to also be marked as `down`.
9. To configure a Generic Routing Encapsulation (GRE) tunnel between the interfaces of two nodes, enter the following command:
   ```
   ifconfig gre0 tunnel 9.3.149.70 9.3.149.121
   ```

   This creates a GRE tunnel between the local interface `9.3.149.70` and the remote interface `9.3.149.121`. The local end of the tunnel is identified by `gre0`.
10. To associate an IP address with the newly created interface, enter the following command:
    ```
    ifconfig gre0 10.10.10.1
    ```
11. To configure NAT on a GRE tunnel, enter the following command:
    ```
    ifconfig gre0 nat toaddr 127.0.0.1 fromport 80 toport 8080
    ```

    In this example, the original destination port of the GRE packet is `80` and the command changes the destination port to `8080` and the destination address to `127.0.0.1`.
12. To configure a GIF tunnel between one-to-many endpoints (one-to-many tunnel), enter the following command:
    ```
    ifconfig gif0 10.10.10.1 netmask 255.255.255.0 tunnel  2000::4612:6995:6c4a:fa6e
    10.10.10.10,2000::4612:6995:6c4a:fa6a 10.10.10.11,2000::4612:6995:6c4a:
    fa6b 15.15.15.1,2000::4612:6995:6c4a:f777
    ```

    The command creates a one-to-many tunnel between the source (`2000::4612:6995:6c4a:fa6e`) and the following 3 targets:
    - `2000::4612:6995:6c4a:fa6a`
    - `2000::4612:6995:6c4a:fa6b`
    - `2000::4612:6995:6c4a:f777`

    The command also configures the `10.10.10.1` IP address to the GIF interface.
13. To configure a one-to-one GIF tunnel between two endpoints, enter the following command:
    ```
    ifconfig gif0 10.10.10.1 netmask 255.255.255.0 tunnel  2000::4612:6995:6c4a:fa6e
    10.10.10.10,2000::4612:6995:6c4a:fa66
    ```

    The command creates a GIF tunnel between the source (`2000::4612:6995:6c4a:fa6e`) and the target (`2000::4612:6995:6c4a:fa66`).
14. The command also configures the 10.10.10.1 IP address to a GIF interface.

Files

Item	Description
/etc/host	Contains the host name database.
/etc/networks	Contains network names.

Related information:

hosts command

networks file format for TCP/IP

TCP/IP routing

Changing the VLAN IDs of a virtual Ethernet adapter

Adding virtual adapters dynamically

---

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.

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. 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

activate command

Item	Description
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 **ikedb** command. 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
   ```

ipaddr 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.

activate command

Item	Description

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

list command

Item	Description
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.

list command

Item **Description**

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

remove command

Item **Description**

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 phase=1 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

Table 1. log

Item	Description
Purpose	Read the ISAKMP daemon log level from **/etc/isamkpd.conf** and start logging at that level.
Syntax	**ike cmd=log**
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: If the log level or the output file name in **/etc/syslog.conf** are changed, the **refresh -s syslogd** command must also be run.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Files

Item	Description
/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:

syslog command

syslog.conf command

syslogd command

ikedb command

Internet Protocol security

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

LDAP supported operations

ikedb –R LDAP **-p** [**-F**]

ikedb –R LDAP **-g** [*policy-name*]

ikedb –R LDAP **-o**

ikedb –R LDAP **-A** <*policy-name*> [-f <xml file name>] [-h ip/host] -C <Dn Name>

ikedb –R LDAP -D <*policy-name*> [-h ip/host] [-F]

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

To use LDAP supported operations, configure the host as an LDAP client.

Item	Description
-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 preshared key would overwrite one that exists in the database. The default is for such put attempts to fail. When an **-R** switch is present, the local entities are overwritten in case the name is a duplicate of a name specified as part of applicable policy on the host in the configuration. **-s** Swaps the local and remote IDs of all tunnels. This flag facilitates importing a tunnel that is generated by a peer system. This flag affects only tunnels. This option is illegal if the remote ID of any tunnel is a group. **-e** *entity-file* Specifies the name of the file that contains 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**. **-R LDAP** The valid value is **LDAP**. When **-p** is used in conjunction with the **-R** switch, the put operation is done by importing the XML configuration file that is associated with the applicable IPSec configuration policy from the LDAP server.
-h	Specifies host name or IP address along with the **-A** flag or the **-D** flag. The IP address can be IPv4 or IPv6.

Item	Description
-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. **-R LDAP** The valid value is **LDAP**. When the **-g** flag is used in conjunction with the**-R** switch, the get operation is done by displaying XML configuration file stored on the LDAP server for the policy that is associated with the local host. If a policy name is also provided, the xml file stored as part of the policy is displayed on stdout.
-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	Used to provide the IPSec certificate used in the associated clients.
-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 the Linux environment, 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 operation directly to the IKE database. If the filename has a hyphen (-), the results are sent to stdout. This flag is invalid if the -R switch is also present on the command line.
-x	Performs an **expunge** operation on the database. This flag empties the database. This flag is invalid if the **-R** flag is also present on the command line.
-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**. When **-R** switch is present, DTD that specifies all the elements and attribute for the XML file allowed to be stored as part of configuration policy on LDAP is sent to **stdout**.
-A	Associates the IP addresses provided with the policy name. If no IP addresses are provided, the first local IPV6 address for the local host is selected and associated with the policy. Policy configuration is enforced by downloading the XML file from LDAP and putting it into the database. The tunnels thus defined are activated. **-f < -path to XML file >** If an XML file is provided, it is stored on the LDAP server as the new XML applicable for the defined policy. If the policy does not exist, this flag is required. **-R LDAP** The valid value is LDAP. This switch must be provided on the command line.

Item	Description
-D	Performs disassociation of configuration policy and IP on LDAP server. This flag is invalid without the **-R** switch. The only valid value for the **R** switch is LDAP. **- F** If the last IP address associated with the specified policy is removed, this switch causes the corresponding policy data (XML configuration file) to be deleted from LDAP server. If this flag is not used, the policy is not deleted from the LDAP server.

Files

Item	Description
/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
   ```
4. To **associate** the host that has the IP address 10.10.10.1 with the configuration policy named Poll with certificate */C=US/O=IBM/CN=test01.austin.ibm.com* with xml file ldap.xml, type:

   ```
   ikedb -R LDAP -A Pol1  -f  ldap.xml  -h 10.10.10.1 -C /C=US/O=IBM/CN=test01.austin.ibm.com
   ```

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 character (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 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

Item	Description
-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.

Item	Description
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

Item	Description
/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 reference:
"make Command" on page 618
Related information:
xmkmf command

imapd Daemon

Purpose

Starts the Internet Message Access Protocol (IMAP) server process.

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

Item	Description
-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

The **imapd** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **imapd** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Files

Item	Description
/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:
pop3d command

---

imapds Daemon

Purpose

Starts the Internet Message Access Protocol (IMAP) server process over TSL/SSL.

Syntax

imapds [**-c**]

Description

The **imapds** 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 **imapds** command with the **inetd** daemon with those descriptors attached to a remote client connection.

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

Flags

Item	Description
-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

The **imapds** daemon is a PAM-enabled application with a service name of *imap*. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the **usw** stanza of **/etc/security/login.cfg**, to PAM_AUTH as the root user.

The authentication mechanisms used when PAM is enabled depend on the configuration for the **imap** service in **/etc/pam.conf**. The **imapds** daemon requires **/etc/pam.conf** entries for the **auth** and **session** module types. Listed below is a recommended configuration in **/etc/pam.conf** for the **imap** service:

```
#
# AIX imap configuration
#
imap auth      required     /usr/lib/security/pam_aix

imap session   required     /usr/lib/security/pam_aix
```

Files

Item	Description
/usr/sbin/imapds	Contains the **imapds** command.
/etc/services	Specifies the file with port assignments for required services. The following entry must be in this file: `imaps  993/tcp  # imap4 protocol over TLS/SSL`

Related information:

pop3ds command

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. IPsec filter rules for this command can be configured using the **genfilt** command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related information:

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** *VolumeGroup*] [**-n**] [**-F**] [**-R**] [**-I**] [**-O**] *PhysicalVolume*

Description

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. If the **importvg** command renames any filesystem log logical volumes, you must manually update any file systems using that log device to know about the renamed device.

Notes:

1. To use this command, you must either have root user authority or be a member of the **system** group.
2. 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 System Management Interface Tool (SMIT) **smit importvg** fast path to run this command.

Flags

Item	Description
-c	This flag is ignored. 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:** • The volume group must not be in an active state on the system executing the **-L** flag. • 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. • 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. • 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.

Item	Description
-n	Causes the volume not to be varied at the completion of the volume group import into the system.
-O	Forces varyon the volume group even if it is varied on in some other node. **Note:** In AIX 61 TL8 and later releases, **varyonvg** command updates the LVM metadata and ODM with varyon state of the volume group. During varyon time, **varyonvg** command reads this data and fails if the volume group is already varied in another node. **Varyoffvg** command resets the varyon state of the volume group during varyoff time. If system crashes before varying off the volume group or the volume group is forced off, then **varyonvg** command will fail after reboot. In this scenario, use **–O** flag to force varyon the volume group.
-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 volume groups of the types big and scalable.
-V *MajorNumber*	Specifies the major number of the imported volume group.
-x	This flag is ignored. Only Enhanced Concurrent Capable volume groups will be created.

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To import the volume group `bkvg` from physical volume `hdisk7`, enter:

   ```
   importvg  -y bkvg hdisk7
   ```

   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 hdisk7
   Node A: varyonvg datavg
   ```

Files

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

Related information:

exportvg command

varyonvg command

Logical volume storage

PowerHA SystemMirror Administration Guide

---

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"mktun Command" on page 839

Related information:

gentun command

chtun command

exptun 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

Item	Description
-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:

Item	Description
`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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 **test cases**, enter:

   ```
   inc     +test cases
   ```

   The system prompts you as follows:

```
Create folder "/home/mary/test cases"?
```

If you wish to create the folder, enter:

```
yes
```

A message similar to the following is displayed:

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

Files

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

Related reference:

"mhmail Command" on page 646

Related information:

scan command

.mh_profile file

Mail applications

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.

Item	Description
-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`
	`}`

Item	Description

-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.

Item	Description
-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.

Item	Description

-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

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

Related information:

cb command

Commands command

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.

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

Files

Item	Description
/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

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

Related reference:
"lookbib Command" on page 261
Related information:
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` file, using SMIT, then the **inetd** daemon will be refreshed automatically and will read the new `/etc/inetd.conf` file. If you change the `/etc/inetd.conf` file using any other editor, run the **refresh -s inetd** or **kill -1** *InetdPID* command to inform the **inetd** daemon of the changes to its configuration file. If you run the **refresh -s inetd** command, the running services continue to run with old configuration until the services terminate and the new services are created.

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

Item	Description
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

Item	Description
-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:

Item	Description
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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related information:

ftpd command

talkd command

inetd.conf file format for TCP/IP

TCP/IP daemons

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 **-l** (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

Item	Description
-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.

Item	Description
-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

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

Related information:

tic command

captoinfo command

terminfo command

---

telinit or init Command

Purpose

Initializes and controls processes.

Syntax

{ **telinit** | **init** } { **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **a** | **b** | **c** | **h** | **Q** | **q** | **S** | **s** | **M** | **m** | **N** }

Description

The **init** command initializes and controls processes. Its primary role is to start processes based on records read from the **/etc/inittab** file. The **/etc/inittab** file usually requests that the **init** command run the **getty** command for each line on which a user can log in. The **init** command controls autonomous processes required by the system.

The process that constitutes the majority of the **init** command's process dispatching activities is **/usr/sbin/getty**. The **/usr/sbin/getty** process initiates individual terminal lines. Other processes typically dispatched by the **init** command are daemons and the shell.

The **telinit** command, which is linked to the **init** command, directs the actions of the **init** command. The **telinit** command takes a one-character argument and signals the **init** command by way of the **kill** subroutine to perform the appropriate action.

The **telinit** command sets the system at a specific run level. A run level is a software configuration that allows only a selected group of processes to exist. The system can be at one of the following run levels:

Item	Description
0-9	Tells the **init** command to place the system in one of the run levels **0-9**. When the **init** command requests a change to run levels **0-9**, it kills all processes at the current run levels and then restarts any processes associated with the new run levels.
0-1	Reserved for the future use of the operating system.
2	Contains all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system. The console device driver is also set to run at all run levels so the system can be operated with only the console active.
3-9	Can be defined according to the user's preferences.
S,s,M,m	Tells the **init** command to enter the maintenance mode. When the system enters maintenance mode from another run level, only the system console is used as the terminal.

The following arguments also serve as directives to the **init** command:

Item	Description
a,b,c,h	Tells the **init** command to process only those records in the **/etc/inittab** file with **a**, **b**, **c**, or **h** in the run level field. These four arguements, **a**, **b**, **c**, and **h**, are not true run levels. They differ from run levels in that the **init** command cannot request the entire system to enter run levels **a, b, c**, or **h**. When the **init** command finds a record in the **/etc/inittab** file with a value of **a**, **b**, **c**, or **h** in the run level field, it starts the process. However, it does not kill any processes at the current run level; processes with a value of **a**, **b**, **c**, or **h** in the run level field are started in addition to the processes already running at the current system run level. Another difference between true run levels and **a**, **b**, **c**, or **h** is that processes started with **a**, **b**, **c**, or **h** are not stopped when the **init** command changes run levels. Three ways stop **a**, **b**, **c**, or **h** processes: • Type **off** in the *Action* field. • Delete the objects entirely. • Use the **init** command to enter maintenance state.
Q,q	Tells the **init** command to re-examine the **/etc/inittab** file.
N	Sends a signal that stops processes from being respawned.

During system startup, after the root file system has been mounted in the pre-initialization process, the following sequence of events occurs:

1. The **init** command is run as the last step of the startup process.
2. The **init** command attempts to read the **/etc/inittab** file.
3. If the **/etc/inittab** file exists, the **init** command attempts to locate an `initdefault` entry in the **/etc/inittab** file.
   a. If the `initdefault` entry exists, the **init** command uses the specified run level as the initial system run level.
   b. If the `initdefault` entry does not exist, the **init** command requests that the user enter a run level from the system console (**/dev/console**).
   c. If the user enters an **S**, **s**, **M** or **m** run level, the **init** command enters maintenance run level. These are the only run levels that do not require a properly formatted **/etc/inittab** file.
4. If the **/etc/inittab** file does not exist, the **init** command places the system in the maintenance run level by default.
5. The **init** command rereads the **/etc/inittab** file every 60 seconds. If the **/etc/inittab** file has changed since the last time the **init** command read it, the new commands in the **/etc/inittab** file are executed during system startup.

When you request the **init** command to change the run level, the **init** command reads the **/etc/inittab** file to identify what processes should exist at the new run level. Then, the **init** command cancels all processes that should not be running at the new level and starts any processes that should be running at the new level.

The processes run by the **init** command for each of these run levels are defined in the **/etc/inittab** file. The run level is changed by having a root user run the **telinit** command, which is linked to the **init** command. This user-run **init** command sends appropriate signals to the original **init** command initiated by the system during startup. The default run level can be changed by modifying the run level for the `initdefault` entry in the **/etc/inittab** file.

In the maintenance run level, the **/dev/console** console terminal is opened for reading and writing. The password for root is prompted. When the root password is entered successfully, the **su** command is invoked. Two ways exist to exit from the maintenance run level:

- If the shell is terminated, the **init** command requests a new run level.

  OR
- The **init** (or **telinit**) command can signal the **init** command and force it to change the run level of the system.

During a system startup attempt, apparent failure of the **init** command to prompt for a new run level (when **initdefault** is maintenance) may be due to the fact that the terminal console device (**/dev/console**) has been switched to a device other than the physical console. If this occurs and you wish to work at the physical console rather than the **/dev/console**, you can force the **init** command to switch to the physical console by pressing the DEL (delete) key at the physical console device.

When the **init** command prompts for a new run level, enter one of the digits **0** through **9** or any of the letters **S**, **s**, **M**, or **m**. If you enter **S**, **s**, **M**, or **m**, the **init** command operates in maintenance mode with the additional result that if control had previously been forced to switch to the physical console, the **/dev/console** file is switched to this device as well. The **init** command generates a message to this effect on the device to which the **/dev/console** file was previously connected.

If you enter a **0** through **9** run level, the **init** command enters the corresponding run level. The **init** command rejects any other input and re-prompts you for the correct input. If this is the first time the **init** command enters any run level other than maintenance, it searches the **/etc/inittab** file for entries with the **boot** or **bootwait** keywords. If the **init** command finds these keywords, it performs the corresponding task, provided the run level entered matches that of the entry. For example, if the **init** command finds the **boot** keyword, it boots the machine. Any special initialization of the system, such as checking and mounting file systems, takes place before any users are allowed on the system. The **init** command then scans the **/etc/inittab** file to find all entries that are processes for that level. It then resumes normal processing of the **/etc/inittab** file.

Run level **2** is defined by default to contain all of the terminal processes and daemons that are run in the multiuser environment. In the multiuser environment, the **/etc/inittab** file is set up so that the **init** command creates a process for each terminal on the system.

For terminal processes, the shell terminates either as a result of an end of file character (EOF) typed explicitly or as the result of disconnection. When the **init** command receives a signal telling it that a process has terminated, it records the fact and the reason it stopped in **/etc/utmp** file and **/var/adm/wtmp** file. The **/var/adm/wtmp** file keeps a history of the processes started.

To start each process in the **/etc/inittab** file, the **init** command waits for one of its descendant processes to stop, for a power fail signal **SIGPWR**, or until the **init** command is signaled by the **init** or **telinit** commands to change the system's run level. When one of the above three conditions occurs, the **init** command re-examines the **/etc/inittab** file. Even if new entries have been added to the **/etc/inittab** file, the **init** command still waits for one of the three conditions to occur. To provide for instantaneous response, re-examine the **/etc/inittab** file by running the **telinit -q** command.

If the **init** command finds that it is continuously running an entry in the **/etc/inittab** file (more than five times in 225 seconds), it assumes that an error in the entry command string exists. It then prints an error message to the console and logs an error in the system error log. After the message is sent, the entry does

not run for 60 seconds. If the error continues to occur, the command will respawn the entry only five times every 240 seconds. The **init** command continues to assume an error occurred until the command does not respond five times in the interval, or until it receives a signal from a user. The **init** command logs an error for only the first occurrence of the error.

When the **init** command is requested to change run levels by the **telinit** command, the **init** command sends a **SIGTERM** signal to all processes that are undefined in the current run level. The **init** command waits 20 seconds before stopping these processes with the **SIGKILL** signal.

If the **init** command receives a **SIGPWR** signal and is not in maintenance mode, it scans the **/etc/inittab** file for special power fail entries. The **init** command invokes the tasks associated with these entries (if the run levels permit) before any further processing takes place. In this way, the **init** command can perform cleanup and recording functions whenever the system experiences a power failure. It is important to note that these power fail entries should not use devices that need to be initialized first.

Environments

Because the **init** command is the ultimate ancestor of every process on the system, every other process on the system inherits the **init** command's environment variables. As part of its initialization sequence, the **init** command reads the **/etc/environment** file and copies any assignments found in that file into the environment passed to all of its subprocesses. Because **init** subprocesses do not run from within a login session, they do not inherit a umask setting from **init**. These processes may set the umask to whatever value they require. A command that is executed by **init** from the **/etc/inittab** file uses **init**'s ulimit values and not the default values as given in **/etc/security/limits**. The result is that a command that is successfully executed from the command line may not execute correctly when invoked by **init**. Any command that has specific **ulimit** requirements should include specific actions to set the **ulimit** values as required.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To request the **init** command to reexamine the **/etc/inittab** file, enter:

   ```
   telinit  q
   ```

2. To request the **init** command to enter maintenance mode, enter:

   ```
   telinit  s
   ```

Files

Item	Description
/etc/inittab	Specifies the **init** command control file.
/etc/utmp	Specifies the record of logged-in users.
/var/adm/wtmp	Specifies the permanent login accounting file.
/sbin/rc.boot	Specifies the pre-initialization command file.
/etc/rc	Specifies the initialization command file.
/etc/environment	Specifies system environment variables.
/dev/console	Specifies the console device driver.

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

Item	Description
-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.

Item	Description
-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

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

Related reference:

"installbsd Command" on page 59

"mv Command" on page 919

Related information:

cp command

strip command

install_all_updates Command

Purpose

Updates installed software to the latest level on media and verifies the current recommended maintenance or technology 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 except in the following situations:

- the new filesets are installed as requisites of other filesets.
- the **/var/adm/ras/bosinst.data** filesets ALL_DEVICES_KERNELS to `yes`.

For installp images, all installp requisites are enforced.

Note:

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 or technology level by using the "**oslevel**" utility and checking with the latest recommended maintenance or technology 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 file system expansions. The "**-n**" will override the install requisite default, and "**-x**" will override the file system expansion default.

6. The following flags apply to installp updates *only*: **-c**, **-n**, **-x**, **-v**, **-S**, and **-V**.
7. Any library or executable program updated by an interim fix or service update which is in use by an active process will not be reflected in that process unless it is restarted. For example, an update that changes the ksh will not have the changes reflected in any ksh processes that are already running. Likewise, an update to the **libc.a** library will not be reflected in any process that is already running. In addition, any process that is using a library and does a **dlopen** operation of the same library after the library has been updated could experience inconsistencies if it is not restarted.
8. If an attempt is made to update a fileset that is locked by the interim fix manager (the **emgr** command), a notice will be displayed indicating which filesets are locked. The **lslpp** command shows that any locked filesets are in the EFIXLOCKED state.
9. If an attempt is made to update a file set that has an installed build date more recent than the build date of the selected fileset, a message will be displayed to indicate this.

Some installed software must ship new installation images instead of service updates in new technology levels or service packs of the AIX operating system. For instance, a new installation image is required if the requisites of the installation image changes. When filesets are updated by using the **smitty update_all** or **install_all_updates** command, the most current version of the fileset is installed irrespective of whether filesets are updated by using the installation image or service update in the software source.

When a new installation image is installed, the history of the fileset in the system, which is the output of the **lslpp -ah <fileset>** command, is reset. The output of the **lslpp -ah <fileset>** command lists the new level of the fileset instead of original installation that was installed and all changes after that installation. . The following examples show the history of the **bos.ecc_client.rte** file before and after the installation image is installed.

- Before a new installation image for the **bos.ecc_client.rte** file is shipped, the following output is displayed:

```
# lslpp -ah bos.ecc_client.rte
       Fileset         Level       Action       Status       Date       Time
       ----------------------------------------------------------------------------
       Path: /usr/lib/objrepos  bos.ecc_client.rte
                       6.1.9.0     COMMIT       COMPLETE     04/26/17     16:49:31
                       6.1.9.0     APPLY        COMPLETE     04/26/17     16:49:31
                       6.1.9.15    APPLY        COMPLETE     04/26/17     21:02:55
                       6.1.9.45    APPLY        COMPLETE     04/27/17     08:11:05
       Path: /etc/objrepos  bos.ecc_client.rte
                       6.1.9.0     COMMIT       COMPLETE     04/26/17     16:49:42
                       6.1.9.0     APPLY        COMPLETE     04/26/17     16:49:42
                       6.1.9.15    APPLY        COMPLETE     04/26/17     21:03:07
                       6.1.9.45    APPLY        COMPLETE     04/27/17     08:11:19
```

- After a new installation image for the **bos.ecc_client.rte** file is shipped and installed on the system, the following output is displayed:

```
# lslpp -ah bos.ecc_client.rte
       Fileset         Level        Action       Status       Date         Time
       ----------------------------------------------------------------------------
Path: /usr/lib/objreposbos.ecc_client.rte
                       6.1.9.100    COMMIT       COMPLETE     04/27/17     09:19:12
                       6.1.9.100    APPLY        COMPLETE     04/27/17     09:19:12

Path: /etc/objreposbos.ecc_client.rte
                       6.1.9.100    COMMIT       COMPLETE     04/27/17     09:19:22
                       6.1.9.100    APPLY        COMPLETE     04/27/17     09:19:22
```

Flags

Item	Description
-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 or technology 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**.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To install all installp updates on device **/dev/cd0** and to verify the current recommended maintenance or technology level, enter:

   ```
   install_all_updates -d /dev/cd0
   ```

2. To update any rpm images on your system, with newer technology levels from the **/images** directory, enter:

   ```
   install_all_updates -d /images -r
   ```

3. To install the latest level of install utilities on device **/dev/cd0** (**bos.rte.install** update), enter:

   ```
   install_all_updates -d /dev/cd0 -i
   ```

Files

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

Related reference:
"installp Command" on page 62
"lslpp Command" on page 432
Related information:
Checking fileset build dates

---

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

Item	Description
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:

configassist command

Configuring command

---

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

Item	Description
-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

Item	Description
$HOME/.mh_profile	Contains the MH user profile.

Related information:
Mail applications

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

Item	Description
-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

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

Related reference:

"install Command" on page 52

"mv Command" on page 919

Related information:

chgrp command

cp command

installios Command

Purpose

Sets up the environment and creates NIM resources from the Virtual I/O Server DVD to install the Virtual I/O logical partition and the Integrated Virtualization Manager.

Syntax

To set up the environment and create NIM resources for installing a Virtual I/O logical partition or Integrated Virtualization Manager:

installios [**-p** *partition_name* **-i** *ipaddrorhostname* **-S** *subnet_mask* **-g** *gateway* **-d** *path* **-s** *system_name* **-r** *profile* [**-n**] [**-P** *speed*] [**-D** *duplex*] [**-l** *language*] [**-L** *location*] [**-V** *vlan_tag*] [**-Y** *vlan_priority*]]

To clean up tasks from the setup process:

installios -u [**-f** | **-U**]

Description

The **installios** command creates NIM resources from the Virtual I/O Server DVD to install a Virtual I/O logical partition and Integrated Virtualization Manager. When invoked on a NIM client, the **-L** flag must be specified with the location of the bos.sysmgt.nim.master fileset. The **installios** command configures the client as a NIM master and creates the resources from the Virtual I/O Server DVD to install the ioserver logical partition or the Integrated Virtualization Manager. After the logical partition or Integrated Virtualization Manager have been installed, the **installios** command can return the NIM master back to its original state by removing the created resources from the DVD or by unconfiguring the NIM master. All of the flags are optional. If no flags are specified, the **installios** wizard runs and the user is prompted to interactively enter the flag information.

Flags

Item	Description
-d *path*	Specifies the path to the installation images (**/dev/cd0** or the path to a system backup of the Virtual I/O Server created by the **backupios** command. The path may also specify a remote NFS-mountable location such as *hostname***:/***path_to_backup*.
-D *duplex*	Specifies duplex (optional). This is the duplex setting with which to configure the network interface of a client. The network interface of the client must support the value of the *duplex* parameter. This value can be `full`, `half`, or `auto`. The default value is `full` if you do not specify this flag.

Item	Description
-f	Forces a cleanup to deallocate and remove resources which are not yet installed on a Virtual I/O logical partition or Integrated Virtualization Manager.
-g *gateway*	Specifies the client gateway (the default gateway that the client will use during network installation of the Virtual I/O Server operating system).
-i *ipaddrorhostname*	Specifies the client IP address or hostname (the IP address or hostname with which the network interface of a client will be configured for network installation of the Virtual I/O Server operating system).
-l *language*	Specifies the language (optional). This is the language in which the license agreement will be displayed before the installation. When the license is viewed, a prompt displays asking if the license is to be accepted. If the prompt is answered with y, then the installation will proceed and the Virtual I/O Server license is automatically accepted after installation. If the prompt is answered with n, the **installios** command exits and the installation does not proceed. If this flag is not specified, the installation proceeds, but the Virtual I/O Server will not be usable until the license is manually accepted after installation.
-L *location*	Specifies the location of the bos.sysmgt.nim.master fileset to configure a client to become a NIM master.
-n	Specifies that the network interface of a client should not be configured. If this flag is specified, the network interface of a client will not be configured with the IP settings that were specified in the flags given to the **installios** command after the installation has completed.
-p *partition_name*	Specifies the partition name. This is the name of the LPAR that will be installed with Virtual I/O Server operating system. This partition must be of type Virtual I/O Server and the partition name must match the name shown on the HMC; the name is not a host name.
-P *speed*	Specifies speed (optional). This is the communication speed to use when configuring the network interface of a client. The network interface of the client must support the value of the *speed* parameter. This value can be `10`, `100`, `1000`, or `auto`. The default value is `100` if you do not specify this flag.
-r *profile*	Specifies the profile name. This is the name of the profile that will contain the hardware resources that will be installed.
-s *system_name*	Specifies the managed system (the name of the managed system maintained by the HMC). This name must match the name shown on the HMC; the name is not a host name.
-S *subnet_mask*	Specifies the client subnet mask (the subnet mask with which the network interface of a client will be configured for network installation of the Virtual I/O Server operating system).
-u	Cleans up the environment to return the NIM master back to its original state.
-U	Unconfigures the NIM master.
-V *vlan_tag*	Specifies the virtual local area network (VLAN) tag identifier (0 to 4094) that is used for tagging Ethernet frames during the network installation for virtual network communication.
-Y *vlan_priority*	Specifies the virtual local area network (VLAN) tag priority (0 to 7) that is used for tagging Ethernet frames during the network installation for virtual network communication.

Exit Status

Item	Description
0	The **installios** command was successful.

Security

You must have root authority to run the **installios** command

Examples

1. To create Virtual I/O resources on a NIM master for installing client 9.3.6.234, type:
   ```
   installios -d /dev/cd0 -i 9.3.6.234 -g 9.3.6.1 -S 255.255.255.0
   ```
2. To create Virtual I/O resources on a NIM client for installing client 9.3.6.234 where /tmp contains the bos.sysmgt.nim.master fileset, type:
   ```
   installios -d /dev/cd0 -i 9.3.6.234 -g 9.3.6.1 -S 255.255.255.0 -L /tmp
   ```
3. To clean up tasks performed while creating Virtual I/O resources, type:
   ```
   installios -u
   ```
4. To clean up tasks performed during the creation of Virtual I/O resources on a logical partition which has not yet been installed, type:
   ```
   installios -u -f
   ```
5. To clean up tasks and unconfigure NIM after creating Virtual I/O resources , type:
   ```
   installios -u -U
   ```

Location

/usr/sbin/installios

Files

Item	Description
/usr/sbin/installios	Contains the **installios** command
/etc/niminfo	Contains variables used by NIM

Related information:

nim_master_setup command

nim command

nimconfig 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 [**-R** *path*] [**-a** | **-a -c** [**-N**]] [**-e***LogFile*] [**-V** *Number*] [**-d***Device*] [**-E**] [**-Y**] [**-b**] [**-S**] [**-B**] [**-D**] [**-I**] [**-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 [**-R** *path*] **-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** *path*] **-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 [**-R** *path*] **-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 [**-R** *path*] **-C** [**-b**] [**-e***LogFile*]

To List All Installable Software on Media

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

Description

Note:

1. The **noclobber** option of the Korn or C shell must be unset in the environment from which an installation is performed.
2. Update all can be accomplished with smitty or with **install_all_updates**.

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 image with a different modification level or a different fix 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 (such as committed, broken, committed with applied updates, and committed with committed updates).

When a fileset update is applied to the system, the update is installed. The current version of that software, during 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. The saved files from all previous versions of the software product are removed from the system, 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: Committing the 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 current version of the software product is changed to the immediate previous version of the 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. 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 are 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.
- You do not have enough local disk space for saving replaced files but you have access to ample disk space on a remote system. In this case, 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 able to reject 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, a warning is given stating that the save directory could not be deleted. In this case, you must 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 canceled. 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, ensure that you back 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 path names that easily and uniquely identify each user's system. For example, you can add the system's host name in the path name.
- 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
#   AIX windows Application Development Toolkit Include F

  X11.adt.lib              4.1.0.0                I  N usr
#   AIX windows Application Development Toolkit Libraries
#
  X11.adt.motif            4.1.0.0                I  N usr
#    AIX windows Application Development Toolkit Motif

#
  X11.adt.bitmaps          4.1.0.0                I  N usr
#    AIX windows Application Development Toolkit Bitmap Fi

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

The field descriptions are as follows:

Item	Description
Fileset Name	Name of the fileset to be installed.
Level	Level of the fileset to be installed.
I/U	The type of package to which the fileset belongs. The fileset can 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 or technology 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 or technology 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 field descriptions are as follows:

Item	Description
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 following 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 any 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* where:

Item	Description
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. Modification level is also called maintenance level or technology level.
ffff	is a numeric field of 1 to 4 digits that represents the fix level.

If a user is installing an installation package from installation media that contains only installation packages it is not necessary to specify the level. More than one software product installation package with different levels does not often exist on the same installation medium. However, 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.

You can use the following example to install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images** directory.

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

Note: If there are duplicate filesets at the same level, **installp** uses 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 is as 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:

1. 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.
2. The **installp** command cannot install a mkinstallp package or bff image that is larger than 2 GB in size. An alternative is to break the bff image into multiple packages that are less than 2 GB in size.
3. If an attempt is made to update a fileset that is locked by the interim fix manager (the **emgr** command), a notice is displayed indicating the filesets that are locked. The **lslpp** command shows that any locked filesets are in the EFIXLOCKED state.
4. If an attempt is made to update a fileset that has an installed build date more recent than the build date of the selected fileset, a message is displayed to indicate this.

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 are displayed 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

Item	Description
-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 must 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 backward compatibility, other flags and parameters can be accepted with **installp -C**, but are ignored because all necessary cleanup is attempted.

Item	Description
-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. If the requisite software products are not in the committed state, the commit fails and error messages are displayed. 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 are also 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. Error messages are displayed on the screen and are 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 displayed showing the pending license agreements associated with the selected filesets. If the **-E** flag is specified with the **-l** flag, output is displayed 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 **-I** 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 must 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:relocatable:build date**

Item	Description
-M*Platform*	Specifies the *Platform* value. Any of the following values can be used to list the installable software packages: **R** Specifies POWER® processor-based platform packages only. **N** Specifies neutral packages, that is, packages that are not restricted to the POWER processor-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.
-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**).
-R *path*	Indicates a user-specified installation location.
-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.

Item	Description
-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) file system 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 not the 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.
-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 must also be specified in the Level as in vv.rr.mmmm.ffff.ppppppp.

Exit Status

Item	Description
0 (zero)	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).
nonzero	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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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. To capture a log file of all output from the **installp** command, use the **script** command 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 the installation of the **bos.net.tcp.client** fileset from the CD using the **installp** command, type:
   ```
   installp -pacgXd /dev/cd0 bos.net.tcp.client
   ```
10. To install TCP/IP and one of its updates that are both contained in the **/usr/sys/inst.images**, use the **installp** command as in the following example.

    A summary report is given at the end of the **installp** command 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 is as 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 processor-based machines, type:

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

12. To update all filesets from a CD that are currently installed on the system, type:

```
lslpp -lc | awk -F ":" '{print $2}' | tail -n +2 > /tmp/lslpp
installp -agXd /dev/cd0 -e /tmp/install.log -f /tmp/lslpp
```

where the **-e** logs the output to the **/tmp/install.log** file.

Files

Item	Description
/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 reference:

"inudocm Command" on page 77

"lppchk Command" on page 315

"lslpp Command" on page 432

Related information:

bffcreate command

Checking fileset build dates

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**] [**-R**]

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.

Notes:

- 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 the `/usr/include/errno.h` file.
- Listing interim fix information is possible only when using the **-f**, **-i**, **-k**, **-q**, **-r**, **-t**, and **-v** flags. You cannot install the interim fixes using the **instfix** command.

Flags

Item	Description
-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. To display filesets that are not installed, the **-v** flag must also be used. Status values are: **-** Down level **=** Correct level **+** Superseded **!** Not installed
-d *Device*	Specifies the input device. Not valid with the **-i** and **-a** flags.
-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 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 might be used to list the fixes for that particular platform. **R** Specifies POWER processor-based platform fixes only. **N** Specifies neutral fixes, that is, fixes that are not restricted to the POWER processor-based platform. **A** Specifies all fixes.
-p	Displays file sets 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.
-R	Specifies the User Specified Installation Location (USIL).
-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*	Limits the search operation to a given type when used with the **-i** flag. Valid types are: **f** fix **p** preventive maintenance **i** interim fix
-v	Specifies verbose mode when used with the **-i** flag. Displays information about each fileset associated with a fix or keyword. Use this flag with the **-i** flag to display filesets that are not installed. An uninstalled fileset is indicated by an `!` (exclamation point).

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 processor-based platform, type:

   ```
   instfix  -T  -MN  -d /dev/cd0
   ```

Files

Item	Description
/usr/sbin/instfix	Contains the **instfix** command.
/usr/lib/objrepos/fix	Specifies the path to the Object Data Manager database.

Related reference:
"installp Command" on page 62

---

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

Item	Description
-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

Item	Description
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**.

Item	Description
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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"installp Command" on page 62
"inurecv Command" on page 80
"inurest Command" on page 81
"inusave Command" on page 84

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

Item	Description
-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

Item	Description
/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 reference:
"installp Command" on page 62
Related information:
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

Item	Description
-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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

“installp Command” on page 62

“lslpp Command” on page 432

Related information:

nim command

Installing optional software products and service updates

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

Item	Description
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

Item	Description
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

Item	Description
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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"installp Command" on page 62

"inusave Command" on page 84

---

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

Item	Description
-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

Item	Description
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

Item	Description
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

Item	Description
$INULIBDIR/lpp.acf	Archive control file.

Related reference:
"installp Command" on page 62
"inucp Command" on page 75
"inurecv Command" on page 80
"inusave Command" on page 84

inurid Command

Purpose

Removes information that is used for the installation of diskless or dataless clients and workload partitions 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:** When you remove **inst_root** directories to save disk space, there are implications to doing so. By removing these directories, the system cannot be used to create workload partitions, or be used as a Shared Product Object Tree (SPOT) server of diskless or dataless clients. Also, after **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 workload partition or SPOT server without reinstalling the entire operating system.

Flags

Item	Description
-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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/lib/instl/inurid	Contains the **inurid** command.

Related reference:
"installp Command" on page 62

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

Item	Description
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

Item	Description
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**:

Item	Description
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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"installp Command" on page 62

"inurecv Command" on page 80

Related information:

ar 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

Item	Description
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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/sys/inst.images	The default directory to create a **.toc** file.
.toc	The file created by this command in the specified directory.

Related reference:

"installp Command" on page 62

Related information:

bffcreate 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.

Exit Status

Item	Description
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

Item	Description
/usr/lpp/msg/$LANG/inuumsg.cat	The message catalog.

Related reference:
"installp Command" on page 62

inuwpar Command

Purpose

Performs software installation tasks in detached workload partitions.

Syntax

/usr/sbin/inuwpar [**-d** *directory* | **-D**] [**-G**] { **-A** | **-f** *wparnamesfile* | **-w** *wparname,...* } *cmdname* [*option ...*]

Description

The **inuwpar** command performs a software installation or maintenance task on all detached workload partitions (WPARs) or named detached WPARs. A detached workload partition is a system workload partition with a writable **/usr** file system or a writable **/opt** file system that is not shared with the global environment.

The **inuwpar** command operates on workload partitions that can be specified in the following ways:

- A comma-separated list of workload partitions that are specified by the **-w** flag.
- A list of workload partitions (one per line) in the file that is specified by the **-f** flag.
- All detached system workload partitions if you specify the **-A** flag.

If you specify the **-G** flag, **inuwpar** runs the installation command first in the global environment.

Restriction: You cannot run the **inuwpar** command on application workload partitions. You cannot successfully run the **inuwpar** command on shared system workload partitions that have read-only **/usr** and **/opt** file systems unless a relocation path is specified to the command.

If you do not specify the **-d** or **-D** flag and the options of the *cmdname* command contains a **-d** *directory* option, the **inuwpar** command attempts to mount that directory into the workload partition environment as the installation device for the command.

See the "Parameters" on page 90 section for all installation commands that can be used with the **inuwpar** command:

Flags

Item	Description
-A	Applies the installation command to all detached system workload partitions.
-d *directory*	Specifies the directory in the WPAR where the installation directory is accessible. By default, the directory is mounted from the installation command into a temporary directory within the WPAR file system. If the options of the installation command contain a **-d** *directory* option, the directory is used as the installation directory for the command.
-D	Specifies that the directory that is used in the installation command is accessible within the WPAR file systems.
-f *wparnamesfile*	Specifies a file containing a list of detached workload partitions to which the installation command is applied.
-G	Runs the installation command within the global environment and the detached system workload partitions.

Item	Description
-w *wparname,...*	Specifies one or more detached workload partitions to which the installation command is applied.

Parameters

Item	Description
cmdname	Specifies the installation command to run. You can specify the following installation commands: • **geninstall** • **install_all_updates** • **installp** • **instfix** • **update_all**
option	Specifies the option to be used with the installation command.

Exit Status

Item	Description
0	The command was able to run on all applicable workload partitions. The exit value does not mean that the return code of the command that has been run on all workload partitions was necessarily 0.
>0	An error occurred.

Examples

1. To install the **bos.games** file set and all of its requisite software from the **/mydev** directory in the global environment and all detached workload partitions, enter the following command:
   ```
   inuwpar -G -A installp -qaXd /mydev bos.games
   ```
2. To install the **bos.games** file set and all of its requisite software from the **/mydev** directory in the global environment and the workload partitions that are listed in **/tmp/wparlist** file , enter the following command:
   ```
   inuwpar -G -f /tmp/wparlist installp -qaXd /mydev bos.games
   ```
3. To install all of the file sets that are associated with fix IX38794 from the **/mydev** directory in workload partitions **wpar1** and **wpar5**, enter the following command:
   ```
   inuwpar -w wpar1,wpar5 instfix -k IX38794 -d /mydev
   ```
4. To update installed software to the latest level from **/mydev** directory in all detached workload partitions, enter the following command:
   ```
   inuwpar -A install_all_updates -d /mydev
   ```

Related reference:

"install_all_updates Command" on page 54

"mkwpar Command" on page 853

Related information:

geninstall command

syncwpar command

invscout Command

Purpose

Surveys the host system for currently installed microcode or Vital Product Data (VPD).

Syntax

invscout [**-c**] **-v** [**-m** *machine_type_and_model*] [**-s** *serial_number*] [**-q**]

invscout [**-u** [*mask*]] [**-e**] [**-r**] [**-m** *machine_type_and_model*] [**-s** *serial_number*] [***-catl microcode_catalog_path***] [**-q**]

invscout [**-U** | **-UF** [*mask*]] [**-e**] [**-m** *machine_type_and_model*] [**-s** *serial_number*] [***-catl microcode_catalog_path***] [***-fl microcode_file_path***] [**-q**]

invscout [**-h** | **-g**]

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. The following table describes the types of data that the Microcode Survey gathers and the files in which it stores the data.

Table 2. The data captured and the files produced by the Microcode Survey

File	Data stored	Associated flag	Display and print methods
Microcode Survey Upload File	A comparison of the gathered microcode levels and the latest levels available.	None. Use the **invscout** command with no flags to create a Microcode Survey Upload File.	Upload to a Web server over the Internet.
Microcode Update Results Formatted Text Report File	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The previous level of microcode that was installed on each device. • The level of microcode that is currently installed on each device. • The latest level of microcode that is available for each device. • For each device, the results of an attempted action to update the microcode to the latest level.	Use the **-U** option to create a Microcode Update Results Formatted Text Report File.	Print or display on a monitor.

Table 2. The data captured and the files produced by the Microcode Survey (continued)

File	Data stored	Associated flag	Display and print methods
Microcode Survey Results Formatted Text Report File	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The level of microcode that is currently installed on each device. • The latest level of microcode that is available for each device. • A suggested action for each device that can be applied according to the downloaded catalog.mic file.	Use the **-u** flag to send the file to the screen from where you invoked the **invscout** command.	Print or display on a monitor.
Microcode Survey Formatted Text Report File **Attention:** The Microcode Survey Formatted Text Report File is deprecated. Use the Microcode Survey Results Formatted Text Report File instead.	Contains a subset of the information recorded in the Microcode Survey Upload File. The subset includes the following information: • Information about the **invscout** execution itself. • The level of microcode that is currently installed on each device.	Use the **-r** flag to send the file to the screen from where you invoked the **invscout** command. **Attention:** The **-r** flag is deprecated. Use the –u option instead.	Print or display on a monitor.

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

Item	Description
-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. **Attention:** This flag is deprecated. Use the –u option instead.
-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 or 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.
-U *mask*	Updates devices with available microcode updates. This flag requires that the following items exist on the system: • A valid microcode catalog file • Valid microcode images Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action

Item	Description
-UF *mask*	Updates devices and the system firmware with available microcode updates. This flag requires that the following items exist on the system: • A valid microcode catalog file • Valid microcode images This flag might reboot the system. Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action
-u *mask*	Generates a formatted text report that identifies hardware for which microcode updates are available. This flag requires that a valid catalog file exists on the system. Valid options for *mask* include any combination of the following values: • L, l: Latest • C, c: Current • P, p: Previous • A, a: Available • O, o: Outcome • D, d: Description • E, e: Effect • S, s: Suggested action
-fl microcode_file_path	Changes the default path name of the microcode files. These files are stored in an .rpm format.
-e	Must be used with the –U, –UF, or -u flags. The -e flag sets *catalog.mic* and retrieves the updates from the fix central database.

Exit Status

This command returns the following exit values:

Item	Description
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

Item	Description
/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/*host*.vup	VPD Survey Upload File. The *host* variable is the host name of the system represented in the file.
/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/invs.mic.con.inp	Microcode Survey Concatenation Input Directory.
/var/adm/invscout/invs.mrp	Microcode Survey Formatted Text Report File.
/var/adm/invscoup/invs.murp	Microcode Update Results Formatted Text Report File. This file identifies hardware updates that have been successfully applied.
/var/adm/invscout/invs.murrp	Microcode Survey Results Formatted Text Report File. This file identifies hardware for which updates are available.
/var/adm/invscout/invs.vpd.con.inp	VPD Survey Concatenation Input Directory
/var/adm/invscout/invscout.log	Log 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/tmp	Holds **invscout** temporary files. All files in this directory are deleted at the start of every execution of this command.

Related reference:
"invscoutd Command"

---

invscoutd Command

Purpose

Launches a permanent Inventory Scout server daemon.

Syntax

invscoutd [**-o**] [**-p** *Portno*] [**-b** *Bufsize*] [**-d** *maxcatsize*] [**-t** *Timeout*] [**-v** *Verblev*]

Description

The **invscoutd** command implements a permanent Inventory Scout server daemon on one machine in the local network of the user. The usual client is a Java applet running in the Web browser of the user, 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)

Note:

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 is echoed if provided, along with everything else. Additional data (DATALEN) is not required but is echoed if present, as is, after the request string. For the ECHO request, DATALEN is 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 is parsed and returned if provided. Additional data (DATALEN) is not required but is parsed and returned if provided; however, any actual additional data beyond the request string is 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) is not accepted and is 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) is not accepted and is 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 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) is not accepted and is 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) is not accepted and is 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 is always returned for every action, except for 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.
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.

Item	Description
-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 value of 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.
-d *maxcatsize*	Changes the maximum microcode catalog file size from the default value of 50000 to a value that you specify.
-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 value of 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:

Item	Description
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 does 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

Item	Description
/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 reference:

"invscout Command" on page 90

ioo Command

Purpose

Manages Input/Output (I/O) tunable parameters.

Syntax

ioo [**-p** | **-r**] [**-y**]{ **-o** *Tunable* [**=***NewValue*] }

ioo [**-p** | **-r**] [**-y**] {**-d** *Tunable*}

ioo [**-p** | **-r**] [**-y**] **-D**

ioo [**-p** | **-r**] [**-F**] **-a**

ioo -h [*Tunable*]

ioo [**-F**] **-L** [*Tunable*]

ioo [**-F**] **-x** [*Tunable*]

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

Description

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

The **ioo** command configures Input/Output (I/O) tuning parameters. This command sets or displays current or next boot values for all I/O 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 can either display the value of a parameter or set a new value for a parameter.

If a process reads sequentially from a file, the values that are specified by the **minpgahead** parameter determine the number of pages to be read ahead when the condition is first detected. The value that is 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 that is processed by the write behind algorithm (**numclust**).

Note: The tunable variables which apply to the entire system might not be modified from within a workload partition.

Understanding the Effect of Changing Tunable Parameters

Misuse of the **ioo** command can cause performance degradation or operating-system failure. Before you start experimenting with the **ioo** command, you must be familiar with Performance overview of the Virtual Memory Manager.

Before you modify any tunable parameter, you must first read about all its characteristics in the Tunable Parameters section, and follow any Refer To pointer, to fully understand its purpose.

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

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

Flags

Item	Description
-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 with **-r**) or permanent (when used 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* must be changed (that is, it is not set to its default value) and is of type **Bosboot** or **Reboot**, or if it is of type **Incremental** and is 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 that must be changed are of type **Bosboot** or **Reboot**, or are of type **Incremental** and were 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* must be changed (the specified value is different from 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 without a *NewValue*, the nextboot value for tunable displays. When **-p** is used without a *NewValue*, a value displays only if the current and next boot values for the *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 that apply to **reboot** values when used with the **-o**, **-d**, or **-D** flags. That is, it 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.
-F	Forces restricted tunable parameters to be displayed when you specify the **-a**, **-L** , or **-x** flag on the command line. If you do not specify the **-F** flag, restricted tunables are not included, unless they are named in association with a display flag (the **-o, -a , -x**, or **-L** flag).
-L [*Tunable*]	Lists the characteristics of one or all tunables, one per line, by 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               196    196    196                                    M
------------------------------------------------------------------------------
recoveryMode            1      1      1      0      1      N/A               D
------------------------------------------------------------------------------
...
where:
    CUR = current value
    DEF = default value
    BOOT = reboot value
    MIN = minimal value
    MAX = maximum value
    UNIT = tunable unit of measure
    TYPE = parameter type: D (for Dynamic), S (for Static), R (for Reboot),
               B (for Bosboot), M (for Mount), I (for Incremental),
               C (for Connect), and d (for Deprecated)
    DEPENDENCIES = list of dependent tunable parameters, one per line
```

Item	Description

-x [*Tunable*]	Lists the characteristics of one or all tunables, one per line, by using the following (spreadsheet) format:
	`tunable,current,default,reboot,min,max,unit,type,{dtunable }`
	
	`where:`
	`current = current value`
	`default = default value`
	`reboot = reboot value`
	`min = minimal value`
	`max = maximum value`
	`unit = tunable unit of measure`
	`type = parameter type: D (for Dynamic), S (for Static), R (for Reboot),`
	`B (for Bosboot), M (for Mount), I (for Incremental),`
	`C (for Connect), and d (for Deprecated)`
	`dtunable = space separated list of dependent tunable parameters`

-y	Suppresses the confirmation prompt before the **bosboot** command is run.

If you modify (by using the **-o**, **-d** or **-D** flags) a restricted tunable parameter, it results in a warning message to warn the user that a tunable parameter of the restricted-use type is modified. If you also specify the **-r** or **-p** flags, you are prompted for confirmation of the change. In addition, at system reboot, the presence of restricted tunable parameters, which are in the **/etc/tunables/nextboot** file, is modified to a value that is different from their default value (by using a command line that specifies the **-r** or **-p** flags). The modification results in an error log entry that identifies the list of these modified tunable parameters.

When you modify a tunable, you can specify a tunable parameter value by using the abbreviations K, M, G, T, P, and E to indicate their correspondent values:

Abbreviation	Power of two
K	2^{10}
M	2^{20}
G	2^{30}
T	2^{40}
P	2^{50}
E	2^{60}

Thus, a tunable value of 1024 might be specified as 1-K.

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

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

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

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

Tunable Parameters Type

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

Item	Description
Dynamic	If the parameter can be changed at any time
Static	If the parameter can never be changed
Reboot	If the parameter can be changed only during reboot
Bosboot	If the parameter can be changed only 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 be incrementally increased, except at boot time
Connect	If changes to the parameter are only effective for future socket connections
Deprecated	If changing this parameter is no longer supported by the current release of AIX.

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

Note: The current set of parameters that is managed by the **ioo** command includes only Static, Dynamic, Mount, and Incremental types.

Compatibility Mode

When running in pre-5.2 compatibility mode (controlled by the **pre520tune** attribute of **sys0**, see **Performance tuning enhancements for AIX 5.2** in the*Performance management*), reboot values for parameters, except those parameters that are of type *Bosboot*, are not 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 embedding calls to tuning commands in scripts that are called during the boot sequence. Parameters of type Reboot can therefore be set without the **-r** flag so that existing scripts continue to work.

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

Tunable Parameters

For default values and range of values for tunables, refer the **ioo** command help (**-h** <*tunable_parameter_name*>).

Item	Description
aio_active	**Purpose:** Indicates whether the AIO kernel extension is used and pinned. **Tuning:** A value of 1 indicates that the AIO kernel extension is used and pinned.
aio_maxreqs	**Purpose:** Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time. **Tuning:** The specified number includes I/O requests that are in progress, as well as those requests that are waiting in queues to be initiated. The maximum number of asynchronous I/O requests cannot be less than the value of AIO_MAX, as defined in the `/usr/include/sys/limits.h` file, but can be greater. It would be appropriate for a system with a high volume of asynchronous I/O to have a maximum number of asynchronous I/O requests larger than AIO_MAX.

Item	Description
aio_maxservers	**Purpose:** Specifies the maximum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) allowed to service slow path I/O requests. **Tuning:** This value is a per cpu value. The value of **maxservers** cannot be less than **minservers**. There can never be more than this many asynchronous I/O requests in progress at one time, so this number limits the possible I/O concurrency.
aio_minservers	**Purpose:** Specifies the minimum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) that remain active to process slow path I/O requests. **Tuning:** This value is a per cpu value. The value of minservers cannot be greater than maxservers. When the kernel extension is loaded, no AIO servers are created regardless of the current or default settings. This value allows a minimal AIO footprint on systems where AIO is never used. As I/O requests are initiated, AIO servers are created to service them until the maximum value allowed by maxservers is reached. Once the minservers value is exceeded, the number of servers does not fall below minservers.
aio_server_inactivity	**Purpose:** Specifies how long an AIO server sleeps without servicing an I/O request. **Tuning:** When this time limit is exceeded, the server exits, unless it causes the number of available servers to fall below **minservers**. In this case the server goes back to sleep. The time the server sleeps in this rare case is the larger of the times that are specified for the current and default values for **server_inactivity**. It is a rare case and indicates that there might be an imbalance between the number of available servers and the amount of I/O.
dk_lbp_enabled	**Purpose:** Allows you to enable or disable the support for the Logical Block Provisioning (thin-provisioning) in the AIX operating system. When disabled, AIX will not attempt to release the blocks that is not used from a thin-provisioned disk. **Tuning:** A value of 0 disables the Logical Block Provisioning (LBP) support. A value of 1 enables the LBP support. The default value is 1.
dk_lbp_num_bufs	**Purpose:** Defines the size for the pool of pre-allocated buffers that are used for the LBP support. **Tuning:** Controls the maximum number of *unmap* requests that can be processed by the disk driver at any given time. The buffer pool is a system-wide resource pool. On any thin-provisioned disk, only one *unmap* request can be active at a time. The default value for this parameter is 64 (buffer). For example, if you have 64 buffers then you have 32 KB (64 buffers x 512 bytes = 32 KB) of total pinned memory. The value of this tunable must be in the range of 1 - 1024.
dk_lbp_buf_size	**Purpose:** Defines the size of each buffer in the LBP buffer pool. The default value is 512 bytes. This value can be changed to 4096 (4K), in which case, blocks can released for those disks that support 4K block size. **Tuning:** The value for this tunable should be same as the largest supported block size by any disk attached with the AIX system.

Item	Description
j2_atimeUpdateSymlink	**Purpose:** If **j2_atimeUpdateSymlink** is set to 1, then the access time of the symbolic link of Enhanced journaled file system (JFS2 or Enhanced JFS) is updated on **readlink**. **Tuning:** A value of 0 indicates that the access time of JFS2 symbolic links is not updated on **readlink**. There is a performance penalty that is associated with turning **j2_atimeUpdateSymlink** on, so this tunable must not be changed unless there is a real need for it. **SUSv3** does not require that access time be updated on **readlink**, however JFS, and many other platforms do update the access time on **readlink**. This tunable is provided for compatibility with JFS and other UNIX conformant systems.
j2_dynamicBufferPreallocation	**Purpose:** Specifies the number of 16-K slabs to preallocate when the file system is running low of **bufstructs**. **Tuning:** A value of 16 represents 256-K. File system does not need remounting. The **bufstructs** for JFS2 are now dynamic; the number of buffers that start on the paging device is controlled by **j2_nBufferPerPagerDevice**, but buffers are allocated and destroyed dynamically after this initial value. If the number of "external pager file system I/Os blocked with no **fsbuf** (from **vmstat -v**) increases, the **j2_dynamicBufferPreallocation** must be increased for that file system, as the I/O load on the file system might be exceeding the speed of preallocation. A value of 0 disables dynamic buffer allocation completely.
j2_inodeCacheSize	**Purpose:** Controls the amount of memory JFS2 uses for the inode cache. **Tuning:** The value does not explicitly indicate the amount that might be used, but is instead a scaling factor; it is used in combination with the size of the main memory to determine the maximum memory usage for the inode cache.
j2_maxPageReadAhead	**Purpose:** Specifies the maximum number of pages to be read ahead when a sequentially accessed file is processed on JFS2. **Tuning:** The difference between **minfree** and **maxfree** must always be equal to or greater than **j2_maxPageReadAhead**. If run time decreases when the value of **j2_maxPageReadAhead** increases, ensure that the other performance of the other applications does not deteriorate.
j2_maxRandomWrite	**Purpose:** Specifies a threshold for random writes to accumulate in RAM before subsequent pages are flushed to disk by the write behind algorithm of JFS2. **Tuning:** The random write behind threshold is on a per-file basis. Useful if too many pages are flushed out by **syncd**.
j2_metadataCacheSize	**Purpose:** Controls the amount of memory Enhanced JFS uses for the metadata cache. **Tuning:** The value does not explicitly indicate the amount that is not used, but is instead a scaling factor; it is used in combination with the size of the main memory to determine the maximum memory usage for the inode cache.
j2_minPageReadAhead	**Purpose:** Specifies the minimum number of pages to be read ahead when processing a sequentially accessed file on Enhanced JFS. **Tuning:** Useful to increase if there are lots of large sequential accesses. Ensure that the performance of the other application does not deteriorate. Value of 0 might be useful if I/O pattern is purely random.

Item	Description
j2_nPagesPerWriteBehindCluster	**Purpose:** Specifies the number of pages, per cluster, that is processed by the write behind algorithm of Enhanced JFS. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is sequential. It might 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 must exceed to be considered as random by the random write behind algorithm of Enhanced JFS. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is random and random write behind is enabled (**j2_maxRandomWrite**).
j2_recoveryMode	**Purpose:** Sets the behavior for recovery from JFS2 write errors. **Tuning:** The default value of 1 indicates that automatic recovery from JFS2 write errors is set. The value of 0 indicates that the file systems remain in a degraded mode until unmounted.
j2_syncByVFS	**Purpose:** Changes the delay between each invocation of sync processing for a JFS2 file system. **Tuning:** This tunable allows JFS2 file systems to be synchronized at a rate that is different from the standard sync daemon period. When this tunable is set to a nonzero value, it is the number of seconds to delay between the iterations of sync processing for each JFS2 file system. By using this tunable, the sync operations can be spread out more than the sync daemon can spread because the sync daemon handles all file systems simultaneously. It also allows changing the number of threads that handle file system sync operations.
j2_syncConcurrency	**Purpose:** Changes the number of threads that are run to sync data to JFS2 file systems. Each thread operates on a file system at a time. **Tuning:** When there are many file systems mounted, it might be necessary to increase this value to get all the file systems handled by the sync operation on a timely basis. **Note:** This value is effective only when the **j2_syncByVFS** tunable parameter is nonzero.
j2_syncDelayReport	**Purpose:** Notifies you if the time required to sync the file systems exceeds a specified number of seconds. **Tuning:** This tunable parameter sets the number of seconds that is allowed to complete sync processing for a file system. If that number is exceeded, a message is generated in the `syslog` file. This message is only informative and does not change any other sync behavior.
j2_syncPageCount	**Purpose:** Sets the maximum number of modified pages of a file that is written to disk by the sync system call in a single operation. **Tuning:** When an application that uses file system caching is run and does large numbers of random writes, it might be necessary to adjust this setting to avoid lengthy delays during sync operations.

Item	Description
j2_syncPageLimit	**Purpose:** Sets the maximum number of times that the sync system call uses **j2_syncPageCount** to limit pages written before increasing that count to allow progress on the sync operation. **Tuning:** This tunable must be set when **j2_syncPageCount** is set and must be increased if the effect of the **j2_syncPageCount** change is not sufficient.
lvm_bufcnt	**Purpose:** Specifies the number of LVM buffers for raw physical I/Os. **Tuning:** Applications performing large writes to striped raw logical volumes are not obtaining the wanted throughput rate. LVM splits large raw I/Os into multiple buffers of 128-K a piece. A 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 might help the throughput of the application. If a system performs larger than 1 MB raw I/Os, it might be useful to increase this value.
maxpgahead	**Purpose:** Specifies the maximum number of pages to be read ahead when a sequentially accessed file is processed. **Tuning:** The value must be a power of two and must be greater than or equal to **minpgahead**. Observe the elapsed execution time of critical sequential-I/O-dependent applications with the time command. Because of limitations in the kernel, do not exceed 512 as the maximum value used. The difference between **minfree**, and **maxfree** must always be equal to or greater than **maxpgahead**. If execution time decreases with higher **maxpgahead**, observe other applications to ensure that their performance does not deteriorate.
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. **Tuning:** The random write behind threshold is on a per-file basis. The maximum value indicates the largest file size, in pages. You can change the value if **vmstat n** shows page out and I/O wait time peaks at regular intervals (usually when the sync daemon is writing pages to disk). It is useful to set this value to 1 or higher if numerous I/O occurs when **syncd** runs. A value of 0 disables random write behind and indicates that random writes stay in RAM until a sync operation. Setting **maxrandwrt** ensures that these writes get flushed to disk before the sync operation occurs. However, it might 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.
numclust	**Purpose:** Specifies the number of 16-K clusters that are processed by the sequential write behind algorithm of the VMM. **Tuning:** Useful to increase if more pages must be kept in RAM before they are scheduled for I/O, when the I/O pattern is sequential. It might be appropriate to increase if striped logical volumes or disk arrays are being used.
numfsbufs	**Purpose:** Specifies the number of file system **bufstructs**. **Tuning:** File system must be remounted. 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 wakes it up, once a **bufstruct** is available. might be appropriate to increase if striped logical volumes or disk arrays are being used.

Item	Description
pd_npages	**Purpose:** Specifies the number of pages that must be deleted in one chunk from RAM when a file is deleted. **Tuning:** The maximum value indicates the largest file size, in pages. Real-time applications that experience sluggish response time while files are being deleted. Tuning this option is only useful for real-time applications. If real-time response is critical, adjusting this option might improve response time by spreading the removal of file pages from RAM more evenly over a workload.
posix_aio_active	**Purpose:** Indicates whether the AIO kernel extension is used and pinned. **Tuning:** A value of 1 indicates that the AIO kernel extension is used and pinned.
posix_aio_maxreqs	**Purpose:** Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time. **Tuning:** The specified number includes I/O requests that are in progress, as well as those requests that are waiting in queues to be initiated. The maximum number of asynchronous I/O requests cannot be less than the value of AIO_MAX, as defined in the `/usr/include/sys/limits.h` file, but can be greater. It would be appropriate for a system with a high volume of asynchronous I/O to have a maximum number of asynchronous I/O requests larger than AIO_MAX.
posix_aio_maxservers	**Purpose:** Specifies the maximum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) allowed to service slow path I/O requests. **Tuning:** This value is a per processor value. The value of **maxservers** cannot be less than **minservers**. There can never be more than this many asynchronous I/O requests in progress at one time, so this number limits the possible I/O concurrency.
posix_aio_minservers	**Purpose:** Specifies the minimum number of AIO servers (kernel processes dedicated to asynchronous I/O processing) that remain active to process slow path I/O requests. **Tuning:** This value is a per cpu value. The value of **minservers** cannot be greater than **maxservers**. When the kernel extension is loaded, no AIO servers are created, regardless of the current or default settings. This handling allows a minimal AIO footprint on systems where AIO is never used. As I/O requests are initiated, AIO servers are created to service them until the maximum allowed by **maxservers** is reached. Once the **minservers** values exceed, the number of servers does not fall below **minservers**.
posix_aio_server_inactivity	**Purpose:** Specifies how long an AIO server sleeps without servicing an I/O request. **Tuning:** When the time limit is exceeded, the server exits, unless it causes the number of available servers to fall below **minservers**. In this case the server goes back to sleep. The time the server sleeps in this rare case is the larger of the times that are specified for the current and default values for **server_inactivity**. It is a rare case and indicates that there might be an imbalance between the number of available servers and the amount of I/O.

Memory Usage and Statistics

To display the file system memory usage, enter the following command:

```
cat /proc/sys/fs/jfs2/memory_usage
```

This returns the metadata cache, the **inode** cache, and the total memory usage in bytes.

To display the file system statistics, enter the following command:

```
cat /proc/sys/fs/jfs2/statistics
```

This returns the number of **icache** hits, **icache** misses, and **icache** activates.

To display the system statistics related to the Logical Block Provision support such as the number of times the out-of-buffer value appears and the number of times the **unmap** operation failed, enter the following command:

```
cat/proc/sys/disk/lbp/statistics
```

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current and reboot value, range, unit, type, and dependencies of all tunable parameters that are managed by the **ioo** command, enter the following command:

   ```
   ioo -L
   ```

2. To list the current, default, and reboot values, range, unit, and type of the j2_recoveryMode tunable parameter, enter the following command:

   ```
   ioo -L j2_recoveryMode
   ```

   The result might be similar to the following output:

   ```
   NAME              CUR    DEF    BOOT   MIN    MAX    UNIT          TYPE
   ------------------------------------------------------------------------
   recoveryMode       1      1       1     0      1      n/a            D
   ------------------------------------------------------------------------
   ```

3. To display help information for the j2_nPagesPerWriteBehindCluster tunable parameter, enter the following command:

   ```
   ioo -h j2_nPagesPerWriteBehindCluster
   ```

4. To set maxrandwrt to 4 after the next reboot, enter the following command:

   ```
   ioo -r -o maxrandwrt=4
   ```

5. To permanently reset all ioo tunable parameters to default, enter the following command:

   ```
   ioo -p -D
   ```

6. To list the reboot value of all ioo parameters, enter the following command:

   ```
   ioo -r -a
   ```

7. To list (spreadsheet format) the current and reboot value, range, unit, type, and dependencies of all tunables parameters that are managed by the **ioo** command, enter the following command:

   ```
   ioo -x
   ```

Related information:

no command

raso command

tuncheck command

Kernel Tuning

iostat Command

Purpose

Reports Central Processing Unit (CPU) statistics, asynchronous input/output (AIO) and input/output statistics for the entire system, adapters, TTY devices, disks CD-ROMs, tapes and file systems.

Syntax

iostat [**-a**] [**-b**] [**-l**] [**-s**] [**-t**] [**-T**] [**-V**] [**-z**] [{ **-A** [**-P**] [**-q** | **-Q**] } | { [**-d** | **-p**] [**-D**] [**-R**] }[**-m**]] [{ **-f** | **-F** } [*filesystems*,...]][**-S** *power*] [**-@** *wparname* | **ALL** | **Global**] [*drives* ...] [*interval*] [*count*]

iostat [**-X** [-o *filename*]] [interval[count]]

Restriction: The **-a, -A, -b, -d, -D, -m, -p, -P, -q, -Q, -R, -t,** and **-z** flags, the *drives* parameter, and the *wparname* parameter are restricted inside workload partitions.

Note: You must set an interval when you are using the **-b** flag. The minimum value of the interval that you can specify is 2 seconds for the **-b** flag. The Block IO statistics need to be enabled using the raso tunable **biostat**. Once the raso tunable is enabled to collect Block IO statistics, the operating system takes couple of second to populate the statistics before it can be reported. Hence, you need to wait for few seconds before you issue the **iostat -b** command, after enabling the Block IO statistic collection.

Description

The **iostat** command is used to monitor system input/output (I/O) devices (physical and logical) that are loaded, by observing the time for which these devices are active. The **iostat** command also generates reports that can be used to change system configuration to better balance the I/O load between file systems, physical volumes, and adapters.

The **iostat** command generates an XML file when the **–X** option is specified.

The **iostat** command generates various utilization and throughput reports based on the options that you specify. On multiprocessor systems, CPU statistics are calculated system-wide as averages among all processors.

A report generated by the **iostat** command consists of system configuration information and various utilization and throughput reports. The system configuration row displays at the start of the **iostat** command and whenever there is a change in monitored configuration. In addition to system configuration, WPAR configuration is also displayed for the WPAR that has enforced resource limits when the **-@** flag is used.

The system configuration and WPAR configuration information includes the following values:

lcpu Indicates the number of logical CPUs.

drives Indicates the number of disks (including CDs). This information is displayed only when adapters, disks, or CDs are monitored.

tapes Indicates the number of tapes. This information is displayed only when adapters or tapes are monitored.

ent Indicates the entitled capacity. This information is displayed only when the partition is running with shared processor.

vdisk Indicates the number of virtual devices. This information is displayed only when adapters, disks, or CDs are monitored.

wpars Indicates the number of active system workload partitions. This information is displayed only when you specify the **-@** flag.

maxserver
Indicates the maximum number of AIO servers that can serve slow-path IOs. This is a system-wide value. It is displayed only if asynchronous I/O is monitored.

cpulim
Indicates the processor-resource limit for a WPAR in terms of processor units. This information is displayed only for WPARs with enforced processor-resource limit.

rset Indicates the resource-set type (regular or exclusive) that is associated with the WPAR. This information is displayed only when there is a resource set that is associated with the WPAR.

The *Interval* parameter specifies the amount of time in seconds between each report. If the *Interval* parameter is not specified, the **iostat** command generates a single report containing statistics for the time since system startup (boot). 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.

Beginning with AIX 5.3, the **iostat** command reports number of physical processors consumed (physc) and the percentage of entitlement consumed (% entc) in Micro-Partitioning® environments. These metrics will only be displayed on Micro-Partitioning environments.

Note: Some system resource is consumed in maintaining disk I/O history for the **iostat** command. Use the **sysconfig** subroutine, or the SMIT to stop history accounting. While the **iostat** command is running for *Count* of iterations and if there is a change in system configuration that affects the output of **iostat** command, it prints a warning message about the configuration change. It then continues the output after printing the updated system configuration information and the header.

If you specify the **-a** flag, the information is displayed in a report in the following order:

- An adapter-header row.
- A line of statistics for the adapter.
- A disk or tape-header row and the statistics of all the disks, CD-ROMs, or tapes connected to the adapter. Such reports are generated for all the disk or tape adapters that are connected to the system.
- A line of statistics for each disk or tape that is configured.

If the *Drive* parameter is specified, only those names specified are displayed. One or more alphabetic or alphanumeric values can be specified for Drives. If you specify the *Drive* parameter, the TTY and CPU reports are displayed and the disk or tape report contains statistics for the specified drives. If a drive name that you specified is not found, the report lists that name and displays the message `Drive Not Found` and gives the report of all the available drives on the system. If you did not configure drives on the system, no disk or tape report is generated.

Restriction: The first character in the *Drive* parameter cannot be numeric.

Tape utilization report is generated only if you specified the **-p** or **-a** flag.

Note: The **-@** option is not supported when executed within a workload partition.

Reports

The **iostat** command generates four types of reports, the TTY and CPU utilization report, the disk/tape utilization report, the file system utilization report, the system throughput report and the adapter throughput report.

Tips:

- When you invoke the **iostat** command with the **-@** *ALL* option, if there is no information related to a workload partition (WPAR) for a metric, a dash (-) is displayed in the place of a value.
- When you invoke the **iostat** command with the **-@** *WparName* option or inside a WPAR, if there is no information related to a workload partition (WPAR) for a metric, that metric is marked with "@" and the system-wide value is displayed for that metric.
- If a metric is not available for that release, a dash (-) is displayed in the place of a value.

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 TTY and CPU utilization 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.
`physc`	Shows the number or the fraction of physical processors consumed, displayed only if the partition is running with shared processor.
`% entc`	Shows the percentage of entitled capacity consumed, which is displayed only if the partition is running with shared processor. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.
`% rc`	Shows the percentage of the consumed processor resource. The information is displayed only for WPARs with enforced processor-resource limit.

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

The method used to compute CPU disk I/O wait time is as follows: The AIX operating system only marks 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/Tape Utilization Report

The second report generated by the **iostat** command is the disk/tape utilization report. By default, the disk utilization report is displayed, and you must specify the **-p** flag to display the tape utilization report.

When you specify the **-m** flag, the path utilization report is displayed.

The disk report provides statistics on a per-physical-disk basis, and tape utilization report provides statistics on a per-tape-basis. The default report has the following format:

Item	Description
`% tm_act`	Indicates the percentage of time the physical disk/tape 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/tape. A transfer is an I/O request to the physical disk/tape. 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.

If you specify the **-D** flag, the report has the following metrics for disk/tape. Extended metrics for disk are displayed by default and users need to specify the **–p** option for tape utilization report:

Metrics related to transfers (xfer):

`% tm_act`	Indicates the percentage of time the physical disk/tape was active (bandwidth utilization for the drive).
`bps`	Indicates the amount of data transferred (read or written) per second to the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.
`tps`	Indicates the number of transfers per second that were issued to the physical disk/tape. A transfer is an I/O request to the physical disk/tape. Multiple logical requests can be combined into a single I/O request to the disk. A transfer is of indeterminate size.
`bread`	Indicates the amount of data read per second, from the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.
`bwrtn`	Indicates the amount of data written per second, to the drive. Different suffixes are used to represent the unit of transfer. Default is in bytes per second.

Read Service Metrics (read):

`rps`	Indicates the number of read transfers per second.
`avgserv`	Indicates the average service time per read transfer. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`minserv`	Indicates the minimum read service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`maxserv`	Indicates the maximum read service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`timeouts`	Indicates the number of read timeouts per second.
`fails`	Indicates the number of failed read requests per second.

Write Service Metrics (write):

`wps`	Indicates the number of write transfers per second.
`avgserv`	Indicates the average service time per write transfer. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`minserv`	Indicates the minimum write service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`maxserv`	Indicates the maximum write service time. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`timeouts`	Indicates the number of write timeouts per second.
`fails`	Indicates the number of failed write requests per second.

Item	Description
Wait Queue Service Metrics (queue):	**Restriction:** These metrics are not applicable for tapes.
`avgtime`	Indicates the average time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`mintime`	Indicates the minimum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`maxtime`	Indicates the maximum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
`avgwqsz`	Indicates the average wait queue size.
`avgsqsz`	Indicates the average service queue size.
`sqfull`	Indicates the number of times the service queue becomes full (that is, the disk is not accepting any more service requests) per second.

Suffix	Description
K	1000 bytes
M	1 000 000 bytes if displayed in xfer metrics. Minutes, if displayed in read/write/wait service metrics.
G	1 000 000 000 bytes.
T	1 000 000 000 000 bytes.
S	Seconds.
H	Hour.

Notes:

- For drives that do not support service time metrics, read, write and wait queue service metrics will not be displayed.
- Coherent Accelerator Processor Interface (CAPI) flash disk I/O generated by the user space programs in super pipe mode is not included in the iostat command output.

Statistics for CD-ROM devices are also reported.

Block IO Device Utilization Report

The Block IO Device Utilization report provides statistics per IO device. The report helps you in analyzing the IO statistics at VMM or filesystem, and disk layers of IO stack. The report also helps you in analyzing the performance of the IO stack. The default report has the following format:

Item	Description
`device`	Indicates the device name.
`rbytes`	Indicates the number of bytes read over the monitoring interval. Default unit is bytes; a suffix will be appended if required (1024 =K, 1024K =M).
`wbytes`	Indicates the number of bytes written over the monitoring interval. Default unit is bytes; a suffix will be appended if required.
`rserv`	Indicates the read service time per read over the monitoring interval. Different suffixes are used to represent unit, default unit is millisecond.
`wserv`	Indicates the write service time per write over the monitoring interval. Different suffixes are used to represent unit, default unit is millisecond.
`rerr`	Indicates the number of read errors over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
`werr`	Indicates the number of write errors over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
`reads`	Indicates the number of read requests over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).
`writes`	Indicates the number of write requests over the monitoring interval. Default unit is numbers; a suffix will be appended if required (1000 = K, 1000K = M, 1000M = G).

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:

Item	Description
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.

Tip: The **-s** flag, when used with the **-@** or **-f** flag, displays logical and physical volume throughput, which corresponds to File Systems and Disks respectively.

Adapter Throughput Report

This report is generated if the **-a** flag is specified. This report provides statistics on an adapter-by-adapter basis (for both physical and virtual adapters). This report has the following format for a physical adapter report:

Item	Description
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.

The virtual adapter's default throughput report has the following format:

Item	Description
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.
bkread	Number of blocks received per second from the hosting server to this adapter.
bkwrtn	Number of blocks per second sent from this adapter to the hosting server.
partition-id	The partition ID of the hosting server, which serves the requests sent by this adapter.

The virtual adapter's extended throughput report (**-D** option) has the following format:

Metrics related to transfers (xfer:)	
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.
bkread	Number of blocks received per second from the hosting server to this adapter.
bkwrtn	Number of blocks per second sent from this adapter to the hosting server.
partition-id	The partition ID of the hosting server, which serves the requests sent by this adapter.

Adapter Read Service Metrics (read:)

rps	Indicates the number of read requests per second.
avgserv	Indicates the average time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
minserv	Indicates the minimum time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxserv	Indicates the maximum time to receive a response from the hosting server for the read request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.

Adapter Write Service Metrics (write:)

wps	Indicates the number of write requests per second.
avgserv	Indicates the average time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
minserv	Indicates the minimum time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxserv	Indicates the maximum time to receive a response from the hosting server for the write request sent. Different suffixes are used to represent the unit of time. Default is in milliseconds.

Adapter Wait Queue Metrics (queue:)

avgtime	Indicates the average time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
mintime	Indicates the minimum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
maxtime	Indicates the maximum time spent by a transfer request in the wait queue. Different suffixes are used to represent the unit of time. Default is in milliseconds.
avgwqsz	Indicates the average wait queue size.
avgsqsz	Indicates the average service queue size.
sqfull	Indicates the number of times the service queue becomes full (that is, the hosting server is not accepting any more service requests) per second.

Suffix	Description
K	1000 bytes.
M	1 000 000 bytes if displayed in xfer metrics. Minutes, if displayed in read/write/wait service metrics.
G	1 000 000 000 bytes.
T	1 000 000 000 000 bytes.
S	Seconds.
H	Hours.

Asynchronous I/O Report

The asynchronous I/O report has the following column headers :

Item	Description
avgc	Average global AIO request count per second for the specified interval.
avfc	Average fastpath request count per second for the specified interval.
maxgc	Maximum global AIO request count since the last time this value was fetched.
maxfc	Maximum fastpath request count since the last time this value was fetched.
maxreqs	Specifies the maximum number of asynchronous I/O requests that can be outstanding at one time.

File System Utilization Report

The file system utilization report provides statistics on a per-file-system basis. The default report has the following format:

Item	Description
`Filesystem`	Indicates the file system name.
`% tm_act`	Indicates the percentage of time the file system is active.
`Kbps`	Indicates the amount of data transferred (read or written) to the file system in KB per second.
`Tps`	Indicates the number of transfers per second that are issued to the file system. A transfer is of indeterminate size.
`Kb_read`	The total number of KBs read.
`Kb_wrtn`	The total number of KBs written.

Important: You must specify the disk names before you invoke the **-f** or **-F** flag. If you specify the **-f** or **-F** flag, separate file system names to be monitored by commas.

Disk Input/Output History

To improve performance, the collection of disk input/output statistics is disabled by default. To enable the collection of this data, enter the following command:

```
chdev -l sys0 -a iostat=true
```

To display the current settings, enter the following command:

```
lsattr -E -l sys0 -a iostat
```

If the collection of disk input/output history is disabled and the **iostat** command is called without an interval, the **iostat** command output displays the message `Disk History Since Boot Not Available` instead of disk statistics.

Flags

Item	Description
-a	Displays the adapter throughput report. The **-a** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified. The **-a** flag is mutually exclusive with the **-f** or **-F** flag.
-A	Displays the legacy asynchronous IO utilization report, and turns off the display of TTY utilization report.
-b	Displays the block I/O device utilization statistics. The **-b** flag is mutually exclusive to all flags, except the **-T** flag.
-d	Turns off the display of TTY utilization report or CPU utilization report. If you do not specify the **-d** or **-p** flag, then by default the **-d** flag is turned on. The **-t** and **-d** flags together turn off both disks and TTY or CPU statistics, allowed only with the **-a** or **-s** flags. The **-d** flag is mutually exclusive with the **-t** flag unless you specify the **-a** or **-s** flag, too. The **-d** flag is mutually exclusive with the **-p** flag unless you specify the **-a** or **-s** flag, too.
-D	Displays the extended tape/drive utilization report. Use the **-D** flag with the **-d** or **-p** flag. The **-D** flag is mutually exclusive with the **-t** flag unless you specify the **-a** or **-s** flag, too. The **-D** flag is mutually exclusive with the **-f** or **-F** flag.
-f	Displays the file system utilization report. The **-f** flag is mutually exclusive with the **-a** or **-D** flag. The **-f** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified.
-F	Displays the file system utilization report, and turns off other utilization reports. The **-F** flag is mutually exclusive with the **-a** or **-D** flag. The **-F** flag can be specified with the **-A** flag, but not when the **-q** or **-Q** flag is specified.
-l	Displays the output in long listing mode.
-m	Displays the path utilization report. The **-m** flag is mutually exclusive with the **-t** flag.
-p	Displays the tape utilization report. The **-p** flag is mutually exclusive with the **-d** flag unless you specify the **-a** or **-s** flag, too. **Note:** Only the Atape device utilization is reported.
-P	Displays the POSIX asynchronous IO utilization report, and turns off the display of TTY utilization report.
-q	Specifies AIO queues and their request counts. The **-q** flag can be specified only with **-A** or **-P** flag.

Item	Description
-Q	Displays a list of all the mounted file systems and the associated queue numbers with their request counts. The **-Q** flag can be specified only with **-A** or **-P** flag.
-R	Specifies that the reset of *min** and *max** values should happen at each interval. The default is to reset the values once when **iostat** is started. The **-R** flag can be specified only with the **-D** flag.
-s	Specifies the system throughput report. You can specify the **-a** flag with the **-A** flag, but not when you have specified the **-q** or **-Q** flag. Inside a workload partition, you can specify the **-s** flag only with the **-f** or **-F** flag.
-S *power*	Displays the processor statistics that are multiplied by a value of 10^{power}. The default value of the *power* parameter is 0. The following fields are scaled: • `% user` • `% sys` • `% idle` • `% iowait` • `physc` • `entc` **Note:** By default, the `%user`, `%sys`, `%idle`, and `%iowait` fields are relative to the processor consumption of a WPAR. When you specify the **-S** flag with a nonzero power, the `%user`, `%sys`, `%idle`, and `%iowait` fields are relative to system-wide processor consumption. **Note:** The value of power can only be between 0 and 3.
-t	Turns off the display of disk utilization report. The **-t** and **-d** flags together turn off both disks and TTY or CPU statistics, allowed only with the **-a** or **-s** flags. The **-t** flag is mutually exclusive with the **-d** flag unless you specify the **-a** or **-s** flag, too. The **-t** flag is mutually exclusive with the **-D** flag unless you specify the **-a** or **-s** flag, too. The **-t** flag is mutually exclusive with the **-m** flag.
-T	Displays the time stamp.
-V	Displays valid nonzero statistics.
-z	Resets the disk input/output statistics. Only root users can use this option.
-@	Reports I/O activities of a workload partition: • Specify **-@ ALL** to display the activity for the global environment and all workload partitions in the system. • Specify the **-@** flag with a list of workload partition names to display the activity for that workload partition. • Specify **-@ Global** to display the activity for the global environment only. • Specify the **-@** flag inside a WPAR to display system-wide statistics along with WPAR statistics. The **-@** flag can be specified only with **–d** and **–D**, **-f** or **-F** flags. All possible combinations of the **-s, -T, -f, -F, -d, -D** and **-l** flags are allowed. **Restriction:** The **-@** flag is mutually exclusive with **-a, -t, -z, -A, -P, -q, -Q**, and the **-m** flag.
-X	Generates the XML output. The default file name is **iostat_DDMMYYHHMM.xml** unless you specify a different file name by using the **–o** option.
-o	Specifies the file name for the XML output.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display a single history since boot report for all TTY, CPU, and Disks, enter the following command:

   ```
   iostat
   ```

2. To display a continuous disk report at two second intervals for the disk with the logical name `disk1`, enter the following command:

   ```
   iostat -d disk1 2
   ```

3. To display six reports at two second intervals for the disk with the logical name `disk1`, enter the following command:

```
iostat disk1 2 6
```

4. To display six reports at two second intervals for all disks, enter the following command:
```
iostat -d 2 6
```
5. To display six reports at two second intervals for three disks named `disk1`, `disk2`, `disk3`, enter the following command:
```
iostat disk1 disk2 disk3 2 6
```
6. To print the System throughput report since boot, enter the following command:
```
iostat -s
```
7. To print the adapter throughput reports at 5-second intervals, enter the following command:
```
iostat -a 5
```
8. To print 10 system and adapter throughput reports at 20-second intervals, with only the TTY and CPU report (no disk reports), enter the following command:
```
iostat -sat 20 10
```
9. To print the system and adapter throughput reports with the disk utilization reports of hdisk0 and hdisk7 every 30 seconds, enter the following command:
```
iostat -sad hdisk0 hdisk7 30
```
10. To display time stamp next to each line of output of **iostat**, enter the following command:
```
iostat -T 60
```
11. To display 6 reports at 2-second intervals on AIO, enter the following command:
```
iostat -A 2 6
```
12. To display AIO statistics since boot for queues associated with all mounted file systems, enter the following command:
```
iostat -A -Q
```
13. To display extended drive report for all disks, enter the following command:
```
iostat -D
```
14. To display extended drive report for all tapes, enter the following command:
```
iostat -Dp
```
15. To display extended drive report for a specific disk, enter the following command:
```
iostat –D hdisk0
```
16. To reset the disk input/output statistics, enter the following command:
```
iostat –z
```
17. To display only file system statistics for all workload partitions, enter the following command:
```
iostat –F -@ ALL
```
18. To display system throughput of all workload partitions along with the system, enter the following command:
```
iostat -f -s -@ ALL
```
19. To display file system statistics that are appended with default O/P, enter the following command:
```
iostat –f
```
20. To display logical and physical system throughput, enter the following command:
```
iostat –s -f
```
21. To display throughput for user-specified drives and file systems, enter the following command:
```
iostat hdisk0 hdisk1 –f /dev/fslv00 /dev/fslv01 /dev/fslv02
```
22. To display the processor statistics that are multiplied by a factor of 10, enter the following command:
```
iostat -S 1
```

File

Item	Description
/usr/bin/iostat	Contains the **iostat** command.

Related information:
vmstat command
iostadd command
Monitoring disk I/O
Input and Output Handling Programmer's Overview

---

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*] [**-@** *WparName*]

ipcrm -r {**-q**|**-m**|**-s**} [**-@** *WparName*] *Name*

ipcrm -r -u [**-o** *Owner*] [**-g** *Group*] [**-@** *WparName*]

Description

The **ipcrm** command removes one or more message queues, semaphore sets, or shared memory identifiers.

Note: The **-@** option is not supported when executed within a workload partition.

Flags

Item	Description
-g *Group*	Restricts the removal to unnamed semaphores matching the group specified.
-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.
-o *Owner*	Restricts the removal to unnamed semaphores matching the owner specified.
-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.
-r	Removes named or unnamed real-time interprocess communication objects. The named real-time object is either a real-time message queue (**-q**), a real-time shared memory (**-m**), or a real-time semaphore (**-s**) and is identified by its *Name*.
-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.
-u	Removes all real-time unnamed semaphores. Using a descriptor on a destroyed unnamed semaphore can result in unspecified behavior.

Item	Description
-@ *WparName*	Removes the specified interprocess-communication construct within workload partition *WparName*.

The **msgctl**, **shmctl**, and **semctl** subroutines provide details of the remove operations. The identifiers and keys can be found by using the **ipcs** command.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To remove the shared memory segment associated with *SharedMemoryID* 18602, enter:

```
ipcrm -m 18602
```

Related reference:

"ipcs Command"

Related information:

msgget command

semctl command

shmctl command

---

ipcs Command

Purpose

Reports interprocess communication facility status.

Syntax

ipcs [**-m**] [**-q**] [**-s**] [**-S**] [**-P**] [**-1**] [**-a** | **-b -c -o -p -r -t**] [**-T**] [**-C** *CoreFile*] [**-N** *Kernel*] [**-X**] [**-@** [*WparName*]]

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.

Item	Description
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 in bytes.

Item	Description
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

Item	Description
-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.
-1	When used with the **-S** 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.
-r	Writes information about real-time interprocess communication objects.
-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.
-X	Prints all available characters of each user name, group name of owner, creator, owner group, creator group instead of truncating to the first 8 characters.
-@ [*WparName*]	Reports the interprocess-communication facility status for workload partitions. If WparName is specified, the status of the interprocess communication facility is displayed for that particular workload partition. If no WparName is specified, the status of the interprocess communication facility is displayed for all active workload partitions. The name of the workload partition associated with the object is displayed. Specify **Global** as the *WparName* to display IPC object information for just that operating system environment, excluding any IPC information for workload partitions hosted from that environment.

Note:

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/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 reference:

"ipcrm Command" on page 121

Related information:

msgsnd command

semop command

Commands command

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

Item	Description
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 reference:

"iptrace Daemon" on page 129

"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*

/usr/sbin/ipreport [**-C**] [**-e**] [**-n**] [**-r**] [**-s**] [**-S**] [**-v**] [**-x**] [**-1**] [**-N**] [**-T**] [**-c count**] [**-j pktnum**] [**-X bytes**] *tracefile*

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

Item	Description
-c *count*	Displays the number of packets.
-C	Validates checksum.
-e	Generates the trace report in EBCDIC format. The default format is ASCII.
-j *pktnum*	Jumps to the packet number specified by the *pktnum* variable.
-n	Includes a packet number to facilitate easy comparison of different output formats.
-N	Does not resolve the names.
-r	Decodes remote procedure call (RPC) packets.
-s	Prepends the protocol specification to every line in a packet.
-S	Generates the input file on a sniffer.
-T	Represents the input file in the **tcpdump** format.
-v	Verbose.
-x	Prints the packets in the hexadecimal format.
-X *bytes*	Limits the hexadecimal dumps to the value determined by the *bytes* variable.
-1	Specifies the compatibility trace generated on the AIX Version 3.1 operating system.

Related reference:

"iptrace Daemon" on page 129

Related information:

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

Item	Description
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 reference:

"imptun Command" on page 35

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

---

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

Item	Description
-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).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

---

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*] [**-B**] [**-Q** [**-V**]] [**-T**] [**-S** *snap_length*] *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 that meeting specific criteria. Packets are traced only between the local host on which the **iptrace** daemon is started 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 that is loaded by the **iptrace** daemon remains active in memory if **iptrace** is stopped in 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 might 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:

1. The file that is specified by the *LogFile* parameter must not exist on an NFS-mounted file system. Specifying an output file on an NFS-mounted file system can cause the **iptrace** daemon to hang. In this case, you might not be able to kill the **iptrace** daemon, thus, requiring that you restart the system.
2. If **iptrace** is killed with **kill -9**, it is required that you issue **iptrace -u** to unload the **bpf** kernel extensions or simply reboot. Sometimes, on a busy system, it is required that you issue **iptrace -u** multiple times because of the possibility that the kernel extension used by **iptrace** is busy processing packets.

3. The **iptrace** command supports **srcmstr** as well and can be started and stopped from the command line. If started from the command line, it can be stopped by using the **kill -9** command.

Flags

Item	Description
-a	Suppresses ARP packets.
-b	Changes the **-d** or **-s** flags to bidirectional mode.
-B	Uses BPF for packet capture. The **iptrace** command when used along with the -B option returns error if the command is run inside the WPAR.
-d *Host*	Records packets that are headed for the destination host-specified by the *Host* variable. The *Host* variable can be a host name or an IP 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 that is specified by the *Interface* variable.
-L *Log_size*	This option causes **iptrace** to log data in such a way 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 that is 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 that is 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.
-Q	Enables filtered system tracing for the recorded packets. After the tracing feature is enabled, the AIX trace daemon is run to record the selected system events that are related to the network communication subsystem. **Note:** The tracing feature uses **Berkeley Packet Filter (BPF)** for packet capture.
-s *Host*	Records packets that come from the source that is host-specified by the *Host* variable. The *Host* variable can be a host name or an IP address in dotted decimal format. If used with the **-b** flag, the **-s** flag records packets both going to and coming from the host that is specified by the *Host* variable.
-S *snap_length*	Specifies the **snap** size (how much of each packet is actually captured from the wire) when you run the **iptrace** daemon with the **-B** flag (the **bpf** support). The command `iptrace -S 1500 /tmp/iptrace.dump` limits captured packet size to 1500 bytes. The default is 80 bytes.
-T	Creates a **tcpdump** compatible dump file. To read the output, use **ipreport -T** or **tcpdump -r**.
-u	Unloads the kernel extension that was loaded by the **iptrace** daemon at startup.
-V	Sets the socket debug flag (the **SO_DEBUG** socket option) and trace level on sockets. This flag must be used along with the **-Q** flag.

Exit Status

The command returns the following exit values:

Item	Description
0	The daemon has run successfully.

Item	Description
1	• No interfaces were found. • The **pcap_open_live** subroutine failed. • The **pcap_datalink** subroutine failed. • The **pcap_lookupnet** subroutine failed. • The **pcap_loop** subroutine failed. • The hostname was not found. • The address was formed incorrectly. • The WPAR did not permit the operation. • The **setpri** subroutine failed. • The **fopen** subroutine failed. • The **fstat** subroutine failed. • The interface is unknown when the daemon looks up the link type.
2	The **fread** subroutine on a trace file failed.
5	• Socket creation failed. • The specified file already exists, but the file is not a trace file.
9	• The protocol is not in the **/etc/protocols** file. • The service is not in the **/etc/services** file. • The daemon failed to load trace extension (**netintf**). • The daemon failed to unload trace extension.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To start the **iptrace** daemon with the System Resource Controller (SRC), enter the following command:

   ```
   startsrc -s iptrace -a "/tmp/nettrace"
   ```

   To stop the **iptrace** daemon with SRC enter the following command:

   ```
   stopsrc -s iptrace
   ```

2. To record packets that come in and go 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 that are 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 that come in and go 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 reference:

"ipreport Command" on page 126

Related information:

protocols command

sodebug command

trace Daemon

ipv6policy Command

Purpose

Configures or displays IPv6 policies for the default address selection that is based on RFC 3484.

Syntax

ipv6policy -add *address prefix precedence label*

ipv6policy -delete *address prefix precedence label*

ipv6policy -show

Description

You can use the **ipv6policy** command to configure IPv6 policies that override the default behavior of algorithms in RFC 3484.

Flags

Item	Description
-add	Adds new IPv6 policies to the system.
-delete	Deletes IPv6 policies from the system.
-show	Displays all existing IPv6 policies that are defined on the system.

Parameters

Item	Description
address	Specifies a valid IPv6 address.
prefix	Specifies an IPv6 prefix (a valid integer) according to RFC 3484.
precedence	Specifies a precedence value (a valid integer) according to RFC 3484.
label	Specifies a label value (a valid integer) according to RFC 3484.

Examples

To add a new ipv6 policy to the system for the address `2001::` with `prefix=16`, `precedence=10`, and `label=20`, enter the following command as a root user:

```
ipv6policy -add 2001:: 16 10 20
```

Related information:

Internet Protocol (IP) Version 6

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

Item	Description
-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

Item	Description
/usr/sbin/isC2host	Contains the **isC2host** command.

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

Item	Description
-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

Item	Description
/usr/sbin/isCChost	Contains the **isCChost** command.

isnstgtd Command

Purpose

Manages an Internet Storage Name Service (iSNS) daemon.

Syntax

isnstgtd -t *targetname* [-d debuglevel]

isnstgtd -s

isnstgtd -h

Description

The **isnstgtd** command manages the iSNS daemon. The iSNS daemon refreshes the iSNS registration with the period set into the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**. It also responds to Heartbeat messages sent by a new iSNS server when it starts.

Use **isnstgtd** command with the -t flag to start the iSNS daemon. You can also start it with SRC through the following command:

```
startsrc -s isnstgtd -a '-t targetname'.
```

To kill the daemon, you have to kill the process or stop it with SRC.

Restriction
Do not run more than one isnstgtd daemons on the same machine.

Requirement
The -t or -s flags are mandatory.

The isnstgtd iSNS daemon can be used only if an iSCSI Target Mode Target is defined in the ODM database. This target must have the reg_policy attribute set to isns or slp&isns to be taken into account.

Note: When the command specifies to start the daemon in debug mode (isnstgtd -t targetname -d debuglevel) with a debuglevel greater than zero, the command is not run as a daemon.

Flags

Item	Description
-t *targetname*	Specifies the ODM defined iSCSI Target Mode Target use for iSNS communication.
-d *level*	Specifies the debug level use by isnstgtd. The debug level is between 0 (important) and 7 (debug).
-s	Prints the iSNS servers configuration (defined in the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**) on stdout with the SMIT menu format.
-h	Display the help: command usage.

Examples

1. To run the command as a daemon for the defined iSCSI target tgt, enter the following command:
   ```
   isnstgtd -t tgt
   ```
2. To run the command in debug mode with all debug traces, enter the following command:
   ```
   isnstgtd -t target -d 7 &
   ```

System Ressource Controler (SRC)

The isnstgtd daemon can also be managed with SRC:

Item	Description
startsrc -s **isnstgtd** -a '-t *targetname* [-d *debuglevel*]'	Used to start the iSNS daemon under SRC control.
stopsrc -s **isnstgtd**	Used to stop the iSNS daemon started with SRC.
refresh -s **isnstgtd**	Used to ask the iSNS daemon under SRC control to refresh its iSNS registration refresh period set in the iSNS configuration stanza file **/etc/tmiscsi/isns_servers**.

Related information:

etc/tmiscsi/isns_servers command

---

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/istat	Contains the **istat** command.

Related information:

jfs/filsys.h file

File systems

Directories command

/etc/passwd file

j

The following AIX commands begin with the letter *j*.

j2edlimit Command

Purpose

Manages quota Limits Classes for JFS2 file systems.

Syntax

To edit Quota Limits Classes:

j2edlimit [**-e**] [**-u** | **-g**] *Filesystem*

To list Quota Limits Classes:

j2edlimit -l [**-u** | **-g**] *Filesystem*

To Set an Existing Limits Class as the Default Limits Class:

j2edlimit -d *LimitsClassID* [**-u** | **-g**] *Filesystem*

To Assign a User or Group to a Limits Class:

j2edlimit -a *LimitsClassID* [**-u** *UserName* | **-g** *GroupName*] *Filesystem*

Description

Quotas are managed in JFS2 file systems through the use of Limits Classes. Each Limits Class has hard and soft limits for disk space and file, and grace periods for exceeding the soft limits. Individual users and groups may be assigned to a Limits Class and are then subject to the quotas defined by that class. Any user or group not assigned to a class is subject to the quotas defined by the default class (Class ID 0). Quota limits for all users or groups in a particular class can be changed by using **j2edlimit** to modify the Limits Class, without having to change or duplicate quotas for each user or group. By default, or when used with the **-e** flag, the **j2edlimit** command edits the User Limits Classes for the file system specified on the command line. When used with the **-g** flag, the **j2edlimit** command edits the Group Limits Classes for the specified file system. The command creates a temporary file that contains the file system's current limits classes, then invokes the **vi** editor (or the editor specified by the EDITOR environment variable) on the temporary file so that the limits classes can be added and modified. When the editor is exited, the command reads the temporary file and modifies the binary quota files to reflect any changes.

Note: If you specify an editor in the EDITOR environment variable, you must use the full pathname of the editor.

Fields displayed in the temporary file are:

Block Hard Limit
The total amount of 1KB blocks the user or group will be allowed to use, including temporary storage during a quota grace period.

© Copyright IBM Corp. 2015, 2017

Block Soft Limit
The number of 1KB blocks the user or group will be allowed to use during normal operations.

File Hard Limit
The total number of files the user or group will be allowed to create, including temporary files created during a quota grace period.

File Soft Limit
The number of files the user or group will be allowed to create during normal operations.

Block Grace Period
Amount of time a user can exceed the Block Soft Limit before it becomes enforced as a hard limit.

File Grace Period
Amount of time a user can exceed the File Soft Limit before it becomes enforced as a hard limit.

Note:

1. A hard limit with a value of 1 indicates that no allocations are permitted. A soft limit with a value of 1, in conjunction with a hard limit with a value of 0, indicates that allocations are permitted only on a temporary basis. Hard or soft limits can be specified in kilobytes (the default), megabytes, or gigabytes.
2. A user can exceed established soft limits for the length of the corresponding grace period. Upon expiration of the grace period, the soft limit is enforced as a hard limit. The grace period can be specified in days, hours, minutes, or seconds. A value of 0 indicates that the default grace period is imposed; a value of 1 second indicates that no grace period is granted.
3. After changing a grace period using the **j2edlimit** command, users who have already reached their old grace period must reduce their file system usage to a level below their soft limits in order to use the new grace period. In the future, when these users exceed their soft limits, the new grace period will be in effect.

Flags

Item	Description
-a	Assigns the User or Group specified by the -u or -g flag to the indicated Limits Class in the file system specified on the command line.
-d	Sets the indicated Limits Class as the default for the file system specified on the command line. By default, or with the **-u** flag, the default is set for User quotas. With the **-g** flag, the default is set for Group quotas.
-e	Edits the Limits Classes for the file system specified on the command line (this is the default operation for the **j2edlimit** command). By default, or with the **-u** flag, the default is set for User quotas. With the **-g** flag, the default is set for Group quotas.
-g	When used with the **-d**, **-l** or optional **-e** flag, performs the operation on the Group Limits Classes for the file system specified on the command line. When used with the **-a** flag, assigns the associated Group to the specified Limits Class. **Note:** If the parameter contains all numbers then it will be treated as a Group ID, and the Group ID will be assigned to the Limits Class.
-l	Lists the Limits Classes for the file system specified on the command line. By default, or with the **-u** flag, User limits classes are listed. With the **-g** flag, Group limits classes are listed. The format of the listing is the same as found in the temporary file when editing Limits Classes.
-u	When used with the **-d**, **-l** or optional **-e** flag, performs the operation on the User Limits Classes for the file system specified on the command line. When used with the **-a** flag, assigns the associated User to the specified Limits Class. **Note:** If the parameter contains all numbers then it will be treated as a User ID, and the User ID will be assigned to the Limits Class.

Security

Access Control: Only the root user can execute this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To edit User Limits Classes for the **/home** file system:
   ```
   j2edlimit /home
   ```
2. To list Group Limits Classes for the **/home** file system:
   ```
   j2edlimit -l -g /home
   ```
3. To set User Limits Class ID 2 as the default for the **/foo** file system:
   ```
   j2edlimit -d2 /foo
   ```
4. To assign user markg to Limits Class ID 1 in the /home file system:
   ```
   j2edlimit -a 1 -u markg /home
   ```

Files

Item	Description
quota.user	Contains usage and Limits information for users.
quota.group	Contains usage and Limits information for groups.
/etc/filesystems	Contains file system names and locations.

Related information:

quota command

quotacheck command

Disk quota system overview

Setting up the disk quota 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:

Item	Description
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

Item	Description
-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:

Item	Description
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:

   ```
   jobs -p %m
   ```

   Using the jobs reported in Example 1, the screen displays the following process ID:

   ```
   465
   ```

Files

Item	Description
/usr/bin/ksh	Contains the Korn shell **jobs** built-in command.
/usr/bin/jobs	Contains the **jobs** command.

Related reference:

"kill Command" on page 162

"ksh Command" on page 173

Related information:

bg 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

Item	Description
-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:

Item	Description
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  -a2 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:

Item	Description
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 -b +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

Item	Description
/usr/bin/join	Contains the **join** command.
/usr/lib/nls/loc/*.src	Contains collation information.

Related information:

comm command

cut command

sort command

Input and output redirection

---

joinvg Command

Purpose

Joins a snapshot volume group back into its original volume group.

Syntax

joinvg [**-f**] *vgname*

Description

Joins a snapshot volume group that was created with the **splitvg** command back into its original volume group. The snapshot volume group is deleted and the disks reactivated in the original volume group. Any stale partitions will be resynchronized by a background process.

Flags

Item	Description
-f *vgname*	Forces the join when disks in the snapshot volume group are missing or removed. The mirror copy on the missing or removed disks will be removed from the original volume group. The *vgname* parameter specifies the original volume group name with the **splitvg** command.

Security

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

Examples

To join the original volume group, **testvg**, with the snapshot volume group **snapvg**, enter the following command:

```
joinvg testvg
```

Files

Item	Description
/usr/sbin	Directory where the **joinvg** command resides.

Related information:

splitvg command

recreatevg command

k

The following AIX commands begin with the letter *k*.

kdb Command

Purpose

Allows for the examining of a system or live dump or a running kernel.

Syntax

kdb -h

kdb [**-c** *CommandFile*] [**-cp**] [**-i** *HeaderFile*] [**-l**] [**-script**] **-w -u** *KernelFile*

kdb [**-c** *CommandFile*] [**-cp**] [**-i** *HeaderFile*] [**-l**] [**-script**] [**-v**] [*SystemImageFile* [*KernelFile* [*KernelModule ...*]]]

kdb [**-c** *CommandFile*] [**-cp**] [**-i** *HeaderFile*] [**-l**] [**-script**] [**-v**] [**-m** *SystemImageFile*] [**-u** *KernelFile*] [**-k** *KernelModule*]

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.

Root permissions are required to use the **kdb** command on the active system because the **/dev/pmem** special file is used. To run the **kdb** command on the active system, type the following:

```
kdb
```

Note: Stack tracing of the current process on a running system does not work.

To invoke the **kdb** command on a system image file, type the following:

```
kdb SystemImageFile
```

When kdb starts, it looks for a **.kdbinit** file in the user's home directory and in the current working directory. If a **.kdbinit** file exists in either of these locations, kdb will execute all of the commands inside the file as if they were entered at the interactive kdb prompt. If a **.kdbinit** file exists in both of these locations, the file in the home directory will be processed first, followed by the file in the current working directory (unless the current directory is the home directory, in which case the file is processed only once).

Flags

© Copyright IBM Corp. 2015, 2017

Item	Description
-c *CommandFile*	Specifies a different name for the startup script file. If this option is used, then kdb searches for the *CommandFile* parameter in the home and current directories instead of the **.kdbinit** file.
-cp	Causes kdb to print out each command in the startup script files as the command is run. This can be used to help debug the **.kdbinit** files, or any other file specified with the **-c** flag. Each command is printed with a plus (+) sign in front of it.
-h	Displays a short help message in regard to command line usage and a brief listing of the available command line options.
-i *HeaderFile*	Makes all of the C structures defined in the *HeaderFile* parameter available for use with the kdb **print** subcommand. This option requires a C compiler to be installed on the system. If the *HeaderFile* variable needs additional **.h** files to compile, these might have to be specified with separate **-i** options as well.
-k *Module*	Instructs kdb to use the specified *Module* parameter as an additional kernel module for resolving symbol definitions not found in the kernel itself. Using this option is equivalent to specifying the kernel module with the *KernelModule* parameter.
-l	Disables the inline pager (that is, the `more (^C to quit) ?` prompt) in kdb. In this case, the **set scroll** subcommand in kdb has no effect, and the inline pager is always disabled regardless of the scroll setting.
-m *Image*	Instructs kdb to use the specified *Image* parameter as the system image file. Using this option is equivalent to specifying the system image file with the *SystemImageFile* parameter.
-script	Disables the inline pager (that is, the `more (^C to quit) ?` prompt) and disables printing of most status information when kdb starts. This option facilitates parsing of the output from the **kdb** command by scripts and other programs that act as a front end for kdb.
-u *Kernel*	Instructs kdb to use the specified *Kernel* as the kernel file for resolving symbol definitions. Using this option is equivalent to specifying the kernel with the *KernelFile* parameter.
-v	Displays a list of all component dump tables (CDTs) in the dump file when the **kdb** command starts. CDTs list the memory regions that are actually included in the dump. If the **kdb** command is used on a live system, this option is ignored.
-w	Examines a kernel file directly instead of a system image. All kdb subcommands which normally display memory locations from the system image file will instead read data directly from *KernelFile*. Subcommands which write memory are not available.

Parameters

Item	Description
KernelFile	Specifies the AIX kernel that the **kdb** command uses to resolve kernel symbol definitions. A kernel file must be available. When examining a dump, the kernel file must be the same as the kernel that was used to take the system or live dump. The default value is **/unix**.
KernelModule	Specifies the file names of any additional kernel modules that kdb uses to resolve symbol definitions not found in the kernel file itself.
SystemImageFile	Specifies the file that contains the system image. The value can indicate a system or live dump, the name of a dump device, or the **/dev/pmem** special file. The default value is **/dev/pmem**.

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 the following:

   ```
   kdb
   ```

   The **kdb** program returns a (0)> prompt and waits for the 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 the following:

   ```
   kdb /var/adm/ras/vmcore.0 /unix
   ```

   The **kdb** program returns a (0)> prompt and waits for the entry of a subcommand.
3. To invoke the **kdb** command using a live dump file named `/var/adm/ras/livedump/trc1.nocomp.200705222009.00` and the kernel file `/unix`, type the following:

   ```
   kdb /var/adm/ras/livedump/trc1.nocomp.200705222009.00
   ```

Note: The default kernel file is **/unix**. Unlike a system dump, in a live dump, only selected data is present. For example, only the kernel thread data for threads explicitly included in the dump is present.

Files

Item	Description
/usr/sbin/kdb	Contains the **kdb** command.
/dev/pmem	Default system image file.
/unix	Default kernel file.

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

Flags Description

Item	Description
-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

Files

Item	Description
/usr/krb5/bin/kdestroy	Contains the **kdestroy** command.
/var/krb5/security/creds/krb5cc_[uid]	Default credentials cache ([uid] is the UID of the user).

Related reference:

"kinit Command" on page 165

"klist Command" on page 167

Related information:

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

Item	Description
-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

Item	Description
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 reference:

"keylist Command" on page 157

"keypasswd Command" on page 160

Related information:

certadd command

certget command

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**X**X**X**), octal (**o***OOO**O**O**O**), and decimal (**d***DDD**D**D**D**) 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” on page 155.

Flags

Item	Description
<*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

Item	Description
/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:

IMInitializeKeymap command

Input Method Overview

National Language Support Overview

---

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

Item	Description
-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 reference:

"keyadd Command" on page 152

"keylist Command" on page 157

"keypasswd Command" on page 160

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

Item	Description
/usr/sbin/keyenvoy	Contains the **keyenvoy** command.

Related reference:

"keyserv Daemon" on page 161

Related information:

Network File System (NFS) Overview for System Management

NIS overview for System Management

Reference command

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

Item	Description
-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

Item	Description
username	Specifies the AIX user whose key labels is going to be queried.

Exit Status

Item	Description
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 reference:

"keypasswd Command" on page 160

"mksecpki Command" on page 815

Related information:

certrevoke command

certverify command

---

keylogin Command

Purpose

Decrypts and stores the user's secret key.

Syntax

keylogin [**-r**]

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.

Flags

Item	Description
-r	Writes unencrypted secret key into a key file. Use the **-r** flag to store the root user's key in **/etc/.rootkey** on a host. Using this command, processes can run as a superuser task to issue authenticated requests. Therefore, processes do not need to explicitly run the **keylogin** command as a superuser task at system startup time.

Files

Item	Description
/etc/publickey	Contains public or secret keys for NIS maps.

Related reference:
"keyserv Daemon" on page 161
Related information:
chkey command
/etc/publickey file
Network File System (NFS) Overview for System Management
NIS Overview for System Management

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

Item	Description
-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 reference:
"keyadd Command" on page 152
"keylist Command" on page 157
"mksecpki Command" on page 815
Related information:
certverify command

---

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

Item	Description
-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

Item	Description
/etc/.rootkey	Stores the encrypted key for the root directory.

Related reference:

"keyenvoy Command" on page 157

Related information:

chssys command

Network File System

Reference command

System Resource Controller

---

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 that 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

Item	Description
-s{*SignalName* | *SignalNumber*}	Specifies the signal as a signal number or a signal name, such as -9 or **KILL** for the **SIGKILL** signal.
-*SignalName*	Specifies a signal name, such as **HUP**.
-*SignalNumber*	Specifies a signal number. **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 that represents 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 that have 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 that are supported by the implementation
-l*ExitStatus*	Lists signal names that are stripped of the common SIG prefix. If *ExitStatus* is a 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:

Item	Description
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 the following command:

   ```
   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 the following command:

   ```
   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 the following command:

```
kill -kill 0
```

This sends signal 9, the **SIGKILL** signal, to all processes that have a process group ID equal to the senders process group ID. Because the shell cannot ignore the **SIGKILL** signal, this command also stops the login shell and logs you off.

4. To stop all processes that you own, enter the following command:

```
kill -9 -1
```

This command sends signal 9, the **SIGKILL** signal, to all processes that are owned by the effective user, even those processes that are started at other workstations 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 the following command:

```
kill  -USR1  1103
```

The name of the **kill** command is misleading because many signals, including **SIGUSR1**, do not stop processes. The action that is 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 **TERM**.

Files

Item	Description
/usr/include/sys/signal.h	Specifies signal names.

Related reference:

"ksh Command" on page 173

Related information:

csh command

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

Item	Description
-	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*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"kill Command" on page 162

Related information:

signal command

kinit Command

Note: The Kerberos ticket life time is calculated taking the DST changes into consideration, as per design. Hence Kerberos tickets issued during DST disabled time, if has validity that spans to reach the DST enabled time or vice versa can have a difference of 1 hour displayed in **klist**.

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**] [**-u**] [**-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

Flags Description

Item	Description
-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.
-u	Specifies that the **kinit** command creates a credentials cache file that is unique to the process. If the **kinit** command is successful, the credentials cache file name includes a unique number (Process Authentication Group or PAG). In AIX Version 5.3 and later, the PAG is generated from an operating system service. The *KRB5CCNAME* environment variable is set to this credentials cache file, and the **kinit** command executes a new shell.

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

Files

Item	Description
/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 reference:

"klist Command"

"kdestroy Command" on page 151

Related information:

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

Flags Description

Item	Description
-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.

Flags Description

Item	Description
-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

Files

Item	Description
/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 reference:

"kinit Command" on page 165

"kdestroy Command" on page 151

Related information:

env command

kmodctrl Command

Purpose

Loads or unloads the kernel extension **/usr/lib/drivers/kmobip6**.

Syntax

kmodctrl [**-k** *kextname*] [**-luq**]

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

Item	Description
-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 reference:

"mobip6ctrl Command" on page 872

"mobip6reqd Daemon" on page 873

Related information:

Mobile IPv6

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

Parameters

Item	Description
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

Files

Item	Description
/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 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:

Item	Description
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

Item	Description
-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:

rlogin command

rshd command

/etc/inetd.conf file

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 reference:

"inetd Daemon" on page 43

Related information:

rsh command

kvalid_user command

/etc/hosts.equiv file

Authentication and the secure rcmds

---

ksh Command

Purpose

Invokes the Korn shell.

Syntax

ksh [**-i**] [{ **+** | **-** } { **a e f h k m n p 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.

Note: The **ksh** wait built in behaves in a manner similar to the **parent wait()** API.

An enhanced 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.

Additionally, a restricted version of the Korn shell, called **rksh**, is available. The restricted Korn shell allows administrators to provide a controlled execution environment for the users.

Flags

Item	Description
-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, unless in the following conditions: • The simple command is contained in a "&&" or "||" list. • The simple command immediately follows "if", "while" or "until". • The simple command is contained in a pipeline following "!". This mode is disabled when profiles are read.
-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.

Item	Description
-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.
-p	Disables the processing of the **$HOME/.profile** file when you use the shell as a login shell.
-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). Using this flag is the same as issuing the **rksh** command.
-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

Item	Description
/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:

rksh command

Korn shell or POSIX shell commands

Enhanced Korn shell (ksh93)

Restricted Korn shell

---

ksh93 Command

Purpose

Invokes the Enhanced Korn shell.

Syntax

ksh93 [+ | - **a b c C e f h i k m n o p r s t u v x BDP**] [+-**R** *file*] [+-**o** *Option*] [*arg*...].

Note: Preceding a flag with + (plus) rather than - (minus) turns off the flag.

Description

The **ksh93** command invokes the Enhanced 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 Enhanced Korn shell has additional features that are not available in the default Korn shell.

Note: The **ksh93** built-in wait behaves in a manner similar to the parent **wait** subroutine.

Flags

Item	Description
-B	Enables brace pattern field generation, and brace group expansion. This is set by default.
-D	Does not execute the script, but results in the set of output strings in double quotation marks preceded by $. These strings are needed for localization of the script in different languages.
-P	If the **-P** or **-o** profiles are present, the shell is called a profile shell.
-a	Exports automatically all subsequent parameters that are defined.
-b	Job completion messages are printed as soon as a background job changes state rather than waiting for the next prompt.
-c	Causes commands to be read from the first argument. The remaining arguments become positional parameters starting from 0.
-C	Prevents existing files from getting truncated when redirection > is used. **O_EXCL** mode is used to create files. Requires >| to truncate a file when **-C** option is used.
-e	If not contained within a || or **&&** command, or following an **if while** or **until** command, or in the pipeline following **!**, 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 generation.
-h	Designates each command as a tracked alias when first encountered. **Note:** The tracked alias feature is now obsolete.
-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.

Item	Description
-k	(obsolete) 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. **Note: ksh93 -n** outputs a warning message for certain syntax. These messages are warnings. Even though these warnings are issued, the execution of the scripts is unaltered. The following are known warning messages: `...` obsolete, use $(...). -a obsolete, use -e. '=' obsolete, use '=='. %s within [[...]] obsolete, use ((...)). set %s obsolete. `{' instead of `in' is obsolete. "obsolete -j must be 1 or 2.
-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. **Braceexpand** Same as the **-B** flag. **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. **interactive** Same as the **-i** flag. **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. **multiline** The built-in editor will use multiple lines of the screen that are longer than the width of the screen. This might not work on all terminals. **noclobber** Same as the **-C** flag. **noexec** Same as the **-n** flag. **noglob** Same as the **-f** flag. **nolog** Prevents function definitions from being saved in the history file.

Item	Description
	notify Same as the **-b** flag.
	nounset Same as the **-u** flag.
	pipefail A pipeline will not complete until all the components of the pipeline are complete. The return value will be that of the last non-zero return value of the last command to fail, or 0 if all return values are 0.
	showme Single commands or pipelines preceded by a ; (semicolon) will be displayed as if the xtrace option were enabled but will not be executed. Otherwise the leading ; (semicolon) will be ignored.
	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 **ksh93** command line.
-p	Disables processing of the **$HOME/.profile** file and uses the **/etc/suid_profile** file instead of the **ENV** file. This mode is on whenever the effective uid (gid) is not equal to the real uid (gid). Turning this off causes the effective uid and gid to be set to the real uid and gid.
-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 path name 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).

Item	Description
-R *File*	A cross reference database is generated when the **-R** *File* option is used. This can be used to find definitions and references for variables and commands by a separate utility.
-s	Causes the **ksh93** 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.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Location

/usr/bin/ksh93

Related information:

env command

wait command

profile command
Enhanced Korn shell (ksh93)
Korn shell or POSIX shell commands

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

Item	Description
-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 reference:

"klist Command" on page 167

l

The following AIX commands begin with the letter *l*.

labcat Command

Purpose

Prints a process's Sensitivity Label (SL) on the banner, and at the top and bottom of each printed page.

Syntax

labcat *files*

labcat [**-P** *pagetype*] [**-U**] [**-p** *lines*] [**-c**] [**-f**] [*files*]

Description

The **labcat** command generates secure binary labels, in human-readable format, for the System V print subsystem with Trusted AIX installed using two modes of operation. This command uses the DIA label-encodings software to produce the labels that appear on the banner and pages.

In general, the **labcat** command parses each printer command sent to the printer. Those commands, which cannot corrupt internal page labeling or affect the permanent state of the printer, is passed through unaffected. Suspect commands are modified if possible, but the **labcat** command resets the printer and then exits with an error code if the use of a particular command is unacceptable, thereby aborting the print job. Thus, dangerous commands are not sent to the printer. The next printer reset (preceding the next print job) re-establishes the printer default state.

The **labcat** command examines the shell environment variable **TERM** (set by the **lpsched** command to indicate the printer type) to determine the language of the printer commands that it receives from standard input. A value of hplaserjet or hplaser implies PCL language (standard configuration); PS, PS-b, or PSR indicates PostScript language (postscript configuration).

Flags

Item	Description
-c	Adds a carriage return to the output.
-f	Indicates that standard input rather than a file is used.
-p *lines*	Indicates the number of lines of text per page.

© Copyright IBM Corp. 2015, 2017

Item	Description
-P *pagetype*	Determines the correct location for the labels to appear at the top and bottom of various size pages. If you do not specify the *pagetype* parameter, the page-type *letter* is assumed. In a postscript configuration, the value for *pagetype* is sent to the printer after the printer is reset, but before the print-job object is sent. In standard configuration, the *pagetype* parameter must be one of the following: • **executive** • **letter** • **legal** • **a4** • **monarcenvelope** • **coml0envelope** • **dlenvelope** • **c5envelope** The *pagetype* parameter is converted into horizontal and vertical positions that are used for the placement of the trusted labels.
-U	Specifies that pages are not to be labeled.
files	Specifies the files to be printed. You can print multiple files. Separate the files with blank spaces.

Exit Status

Item	Description
0	The command completed successfully.
-1	The command failed.

Error codes

The **labcat** command returns a failure under one of the following conditions:

- It cannot audit the event.
- The page length or page type is not valid.
- It cannot open the label-encodings file.
- It cannot open the file to be printed.

labck Command

Purpose

Checks for the consistency of the label-encodings file.

Syntax

labck [**-l**] [**-f** *encodings_file*]

labck [{ **-c** | **-r** } *encodings_file*]

Description

You can use the **labck** command to verify that a label-encodings file is internally consistent. When specified without any flags, the **labck** command verifies the consistency of the system-default, label-encodings file and no message is displayed if the file is proper.

Flags

Item	Description
-l	Lists the system high sensitivity label, system low sensitivity label, system high integrity label, and system low integrity label as defined in the label-encodings file.
-f *encodings_file*	Uses the value that you specify for *encodings_file* instead of the system-default, label-encodings file.
-c *encodings_file*	Copies the contents of the system-default, label-encodings file into the *encodings_file* that you specify. If the file already exists, the command exits with an error.
-r *encodings_file*	Replaces the contents of the system-default, label-encodings file with the contents in the file that you specify using *encodings_file*. The contents are replaced only if the file name that you specify is a valid label-encodings file.

Security

Only authorized users can run the **labck** command.

Item	Description
aix.mls.lef	Required to perform the above operations on the label encodings file.

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Exit Status

The **labck** command returns the following exit values:

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To check the consistency of the system-default, label-encodings file, enter the following command:
   ```
   labck
   ```
2. To check the consistency of the system-default, label-encodings file and print the system the high and low labels, enter the following command:
   ```
   labck -l
   ```
3. To check the consistency of a label-encodings file that is stored in the current directory, enter the following command:
   ```
   labck –f ./labelencodingsfile
   ```
4. To copy the system-default, label-encodings file to a file with the name of **/tmp/lef**, enter the following command:
   ```
   labck –c /tmp/lef
   ```
5. To replace the contents of system-default, label-encodings file with the contents of the **/tmp/lef** file, enter the following command:
   ```
   labck –r /tmp/lef
   ```

Files

Item	Description
/usr/sbin/labck	Contains the **labck** command.
/etc/security/enc/LabelEncodings	System default label encodings file.

Related information:

setsyslab command

getsyslab command

Trusted AIX

last Command

Purpose

Displays information about previous logins.

Syntax

last [**-X**] [**-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)
```

Flags

Item	Description
-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).
-X	Prints all available characters of each user name instead of truncating to the first 8 characters.

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
   ```
5. To display all the recorded logins and logoffs without truncating the user name, type:
   ```
   last -X
   ```

Files

Item	Description
/usr/bin/last	Contains the **last** command.
/var/adm/wtmp	Contains connect-time accounting data, including login, logoff, and shutdown records.

Related reference:

"lastlogin Command" on page 187

"kill Command" on page 162

Related information:

acctcon1 , accton2

System accounting

Setting up an accounting subsystem

---

lastcomm Command

Purpose

Displays information about the last commands executed.

Syntax

lastcomm [**-X**][*Command*] [*Name*] [*Terminal*]

Description

The **lastcomm** command displays information, in reverse chronological order, about all previously executed commands that are still recorded in the **/var/adm/pacct** summary file. 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:

Item	Description
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.

Flags

Item	Description
-X	Prints all available characters of each user name instead of truncating to the first 8 characters.

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
   ```

3. To display information about all previously executed commands recorded in the **/var/adm/pacct** file without truncating the user name, enter:

   ```
   lastcomm -X
   ```

Files

Item	Description
/usr/bin/lastcomm	Contains the **lastcomm** command.
/var/adm/pacct	The directory that contains the current accounting summary files.

Related information:

acctcms command

System accounting

Setting up an accounting subsystem

lastlogin Command

Purpose

Reports the last login date for each user on the system.

Syntax

/usr/sbin/acct/lastlogin [**-X**]

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.

Flags

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. This flag will also cause the **lastlogin** command to write to the **/var/adm/acct/sumx/loginlog** file instead of the **/var/adm/acct/sum/loginlog** file.

Security

Access Control: This command should grant execute (x) access only to members of the ADM group.

Files

Item	Description
/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:

runacct command

cron command

System accounting

Setting up an accounting subsystem

lbxproxy Command

Purpose

Low BandWidth X proxy.

Syntax

lbxproxy [**:<***display*>] [**-help**] [**-display** *Display*] [**-motion** *Number*] [**-terminate** | **-reset**] [**-reconnect**] [**-I**] [**-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

Item	Description
-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.
-I	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**] [**-V**] [**-z**] [**-Z***String*] ... [**-b***Option*] ... [**-L***Directory*] ... { **-f***FileID* ... **-l***Name* ... *InputFile* ... }

or

ld -bsvr4 [**-d**[**y** | **n**]] [**-D** *Number*] [**-e** *Label*] [**-G**] [**-H***Number*] [**-K**] [**-m**] [**-M**] [**-o***Name*] [**-r**] [**-R** *Path*] [**-s**] [**-S***Number*] [**-T***Number*] [**-u** *Name*] ... [**-v**] [**-V**] [**-z** [**defs** | **nodefs**]] [**-z multidefs**] [**-z** [**text** | **nowarntext** | **warntext**]]] [**-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 210.

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.cfg** 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 206.

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 when the **rtl** option is used, a **lib***Name***.so** file to be searched for in several directories. These search directories include any directories that are 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:

Item	Description
_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

Note: When the **-bsvr4** option is present on the **ld** command line, the **-R** and **-z** options are redefined.

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.

Item	Description
-b*Option*	Sets special processing options. This flag can be repeated. For more information on these options, see "Options (-bOptions)" on page 195.
-d [**y** | **n**]	When **-dy** is specified, **ld** uses dynamic linking; this option is equivalent to the **-b so** option. When **-dn** is specified, **ld** uses static linking; this option is equivalent to the **-b nso** option. The default is **-dy**. This option is valid only when the **-bsvr4** option is specified.
-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. If both the **-D** and **-bpD** flags are specified, the latter flag takes precedence. **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.

Item	Description
-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*	Processes the ***libName*.a** file. In dynamic mode, with the **rtl** option, processes the ***libName*.a** or ***libName*.so** 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 with the **rtl** option, the first directory containing either ***libName*.so** or ***libName*.a** satisfies the search. If both files are found in the same directory, ***libName*.so** is used. You can repeat this flag. For more information about dynamic mode, see "Run-time Linking" on page 206. **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**, **-bnolibpath**, or **-bsvr4** 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.)
-R *Path*	Valid only when the **-bsvr4** option is present on the **ld** command line. It defines a colon-separated list of directories used to specify library search directories to the runtime linker. *Path*, if present and not NULL, is recorded in the output file's loader section. Then it is used when linking an executable with shared libraries at runtime. Multiple instances of this option are concatenated together with each *Path* separated by a colon.
-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 Overview3" in *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**. If both the **-T** and **-bpT** flags are specified, the latter flag takes precedence. **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. **Attention:** If *Number* is `0x1000xxxxxxx` and the program linked is a 64-bit program, the system loader loads the executable text into memory starting from the segment at the address `0x10000000000`.
-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.
-V	Writes the version string of **ld** to standard error (stderr).
-z	In the absence of the **-b svr4** option, functions the same as the **-K** flag.
-z defs	Forces a fatal error if any undefined symbols remain at the end of the link. This is the default when an executable is built. It is also useful when building a shared library to assure that the object is self-contained, that is, that all its symbolic references are resolved internally. This option is valid only when the **-b svr4** option is specified. It is equivalent to **-b ernotok** option.

Item	Description
-z nodefs	Allows undefined symbols. This is the default when a shared library is built. When used with executables, the behavior of references to such undefined symbols is unspecified. This option is valid only when the **-b svr4** option is specified. It is equivalent to **-b erok** option.
-z multidefs	Allows multiple symbol definitions. By default, multiple symbol definitions occurring between relocatable objects (**.o** files) will result in a fatal error condition. This option suppresses the error condition and allows the first symbol definition to be taken. This option is valid only when the **-b svr4** option is specified.
-z text	In dynamic mode only, forces a fatal error if any relocations against the **.text** section remain. This option is valid only when the **-b svr4** option is specified.
-z nowarntext	In dynamic mode only, allows relocations against all mappable sections, including the **.text** section. This is the default when building a shared library. This option is valid only when the **-b svr4** option is specified.
-z warntext	In dynamic mode only, warns if any relocations against the **.text** section remain. This is the default when building an executable. This option is valid only when the **-b svr4** option is specified.
-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.

Note:

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.

Item	Description
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.
bigtls	Generates extra code if the size of thread-local storage in the output object or program is larger than 64 KB and a compiler was used that generates direct references to local-exec or local-dynamic thread-local variables. Extra code is needed for every direct reference to a thread-local variable that cannot be addressed with a 16-bit offset. Because a program containing extra code might have poor performance, it is better to reduce the number of thread-local variables using direct references than to use the option. The default option is the **nobigtls** option.
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 209.
caps	Processes all external symbols in uppercase. The default is the **asis** option.

Item	Description
cdtors[**:**[*incl*][**:**[*nnn*] [**:***order*]]]	The linker gathers information about C++ static constructor or destructor functions and saves this information in the output file. The *incl* suboption tells the linker which archive members to search when creating the saved information. The possible values are: **all** Searches all members of all archives for constructor or destructor functions. This is the default. **mbr** Searches for constructor or destructor functions only if the member is included in the output file. Using the **mbr** value is equivalent to using the **-qtwolink** and **-bsvr4** options. **csect** Searches for the constructor or destructor functions only in **csects** included from the archive. Using the **csect** value is equivalent to using the **-qtwolink** option without the **-bsvr4** option. The *nnn* suboption specifies the priority of the output module. This priority is used to control the order in which modules are initialized, in case a program loads multiple modules at the same time. (The priority is ignored if the output file is a program and not a shared object.) The priority can be in the range from -2^{31} to $2^{31}-1$. The default priority is 0. Values in the range from -2^{31} to $-2^{31}+1023$ are reserved for C++ runtime initialization. The *order* suboption specifies the order in which individual constructor or destructor functions are called, for all functions with the same priority. The possible values are: **s** Sorts in an arbitrary order based on function names. This is the default, and is compatible with the order that the XL C++ compiler uses. **c** Sorts in link order. Constructor or destructor functions in the first input file are initialized first. In archives, functions in the first member are initialized first. **r** Sorts in reverse link order. Constructor or destructor functions in the last input file are initialized first. In archives, functions in the last member are initialized first. You can specify this option multiple times, but the last suboption that you specify is used. An unspecified suboption does not affect the current or default value. For example, **-bcdtors:csect:20:s -bcdtors:::r** is the same as **-bcdtors:csect::20:r**. The default is **-bnocdtors**. If **-bcdtors** is specified, this is equivalent to **-bcdtors:all:0:s**. **Note:** 1. If necessary, the XL C++ compiler produces this option automatically. 2. Functions specified with the **-binitfini** option are invoked independently of static constructor or destructor functions. For more information about the order of initialization, see the **dlopen** subroutine.
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.

Item	Description
D: *Number* [*/dsa*] or **maxdata:***Number*[*/dsa*]	Sets the maximum size (in bytes) that is 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 increase the soft data ulimit, if required. The default value is 0. When this option is used, the specified number of bytes are reserved for the user data area. The program might not explicitly map objects, by 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 that are running under Large Program Support and `0xD0000000` for programs that are running under Very Large Program Support. See "Large Program Support Overview" in *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 specified **maxdata** value. 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.
datapsize:*psize*	Requests *psize* page sizes in bytes for data. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b datapsize:16k` or `-b datapsize:0x4000` will request 0x4000 for data and set the `F_VARPG` bit in the **XCOFF** header.
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. When both are in effect, preference is given to **.so** instead of **.a** when present in same directory. When you specify the **rtl** option and static mode is in effect, files ending in **.a** are processed.
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. When the **svr4** option is used, the **E:***FileID* option cancels any **expall** or **expfull** options.
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.

Item	Description
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 or using the **expfull** option. 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**.
expfull	Exports all global symbols other than imported symbols. Exported global symbols include unreferenced symbols defined in archive members, symbols beginning with an underscore (_), and the module's entry point. This option does not affect symbols exported by the **autoexp** option. The default is **noexpfull** unless the **svr4** option is used.
export:*FileID*	Functions the same as the **E:***FileID* option.
f	Functions the same as the **ernotok** option.
forceimprw	Forces read-only CSECTs that contain references to imported symbols to become read-write. The default is **noforceimprw**.
forkpolicy:*policy*	Sets the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, when linking a 64-bit program. If *policy* is **cor**, the _AOUT_FORK_COR flag is also set, requesting the use of the copy-on-reference forktree policy when the program is run. If *policy* is **cow**, the _AOUT_FORK_COR flag is reset, requesting the use of the copy-on-write forktree policy when the program is run. When linking a 32-bit program, this flag is ignored. The default is **noforkpolicy**.
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.

Item	Description

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:
	1. The priorities in the following inclusive ranges are reserved:
	`-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`
	2. Functions specified with the **-binitfini** option are invoked independently of static constructor or destructor functions. For more information about the order of initialization, see the **dlopen** subroutine.

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 *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.
lpdata	Sets the `F_LPDATA` bit in the **XCOFF** header of the executable file. When this bit is set, the process is going to request large pages for its data.

Item	Description
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**. **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.
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 209.
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.
nobigtls	Generates a severe error message if the compiler generates direct references to thread-local variables and the relocation to one of the variables overflows because the size of the thread-local storage is greater than 64 KB. If an output file is produced, it will not run correctly. The **nobigtls** option is the default option.
nobigtoc	Generates a severe error message if the size of the TOC is greater than 64 KB. 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.
nocdtors	Does not gather static constructor or destructor functions. This is the default.
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.
noexpfull	Does not export symbols unless you list them in an export file or you export them with the **autoexp** option. This is the default, unless the **svr4** option is used.
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. This warning is suppressed when the **svr4** option is used.
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**.
noforceimprw	Allows read-only CSECTs to reference imported symbols. This is the default.
noforkpolicy	Clears the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, when linking a 64-bit program. The default forktree policy is used, unless a forktree policy is specified with the VMM_CNTRL environment variable. When linking a 32-bit program, this flag is ignored. This is the default.
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.

Item	Description
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.
nolpdata	Clears the `F_LPDATA` bit in the **XCOFF** header of the executable file. When this bit is not set, the process is going to use small (regular) pages for its data.
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.
noorder_file	Does not map symbols in a specified order. This flag negates the effect of a previous **-border_file** flag. This is the default.
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 unless the **svr4** option is used.
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 unless the **svr4** option is used.
norwexec	Specifies that if the system's **sed_config** setting is not off, the process' private data areas will have non-execute permission.
noshrsymtab	Prevents the _AOUT_SHR_SYMTAB flag from being set in the output object. This is the default state.
nostabsplit	Prevents the debug section to be written to an alternate output file with the extension *.stab*. This is the default setting.
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.

Item	Description
nosymbolic	Assigns the **nosymbolic** attribute to most symbols exported without an explicit attribute. For more information, see “Attributes of Exported Symbols” on page 208. 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.
notmprelname	The binder does not check for general instantiations. **Note:** This option is only needed for 32-bit mode. This option is ignored when building 64-bit objects.
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.
noweaklocal	Resolves weak symbols using normal search order. This option overrides the **weaklocal** option. It is the default option.
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.
order_file:*FileID*	Maps symbols that are listed in *FileID* in the specified order. The symbols that are listed in the file are mapped before other symbols of the same storage-mapping class. Function names that are specified in the file must start with a dot because a function name without a dot denotes a function descriptor.
order:*Specification*	Controls the order in which some symbols are mapped in the output file. The specifications can be: **toc:fileref** If the **-border_file** flag is used, then the TOC symbols that are referenced by any function listed in the order file, are mapped before other TOC symbols. Otherwise, this specification is ignored. **toc:nofileref** Does not order TOC symbols that are based on the order file. **notoc** Does not order TOC symbols in any special way.
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. **Note:** If both the **-bpD** and **-D** flags are specified, the latter flag takes precedence.
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`. **Note:** If both the **-bpT** and **-T** flags are specified, the latter flag takes precedence. See the **-T** flag for additional information.
quiet	Does not write binder subcommands and their results to standard output. This is the default.

Item	Description
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.
ras	Sets a flag in the output module's auxiliary header to signify that the module is both storage key safe and recovery safe. For more information abput how to make a kernel extension be key safe and recovery safe, see*Kernel Extensions and Device Support Programming Concepts* .
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.

Item	Description

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 an import file with the following lines:
	`# autoload`
	`#! (foo.o)`
	
	and adding the import 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**.
	
	If the first line of an import file begins with `#!` (number sign, 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.

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 unless the **svr4** option is used.
rwexec	Specifies that the execute permissions of the process' private data areas will be determined according to the system's **sed_config** setting. This is the default.
rwexec_must	Specifies that the process' private data areas will have execute permission, regardless of the system's **sed_config** setting.
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 209.
shared	Functions the same as the **dynamic** option.
shrsymtab	In 64-bit mode, sets the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 64-bit program, a shared symbol table is created when the program is run. If *Flag* is a 64-bit object but not a 64-bit program, the _AOUT_SHR_SYMTAB flag can be set, but has no effect at runtime. In 32-bit mode, this flag is ignored. The default is **noshrsymtab**.
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.
stabsplit	Causes the debug section to be written to an alternate output file with the extension *.stab*.

Item	Description
stackpsize:*psize*	Requests *psize* page sizes in bytes for process main thread stack. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b stackpsize:16k` or `-b stackpsize:0x4000` will request 0x4000 for process main thread stack and set the `F_VARPG` bit in the **XCOFF** header.
static	Causes the linker to process subsequent shared objects in static mode. In static mode, shared objects are statically linked in the output file.
svr4	This option changes the meaning of some other options on the command line and the standard behavior of the linker. It has the following effect on the linker: • **-b rtl** is set • **-b rtllib** is set only when building an executable or if not set explicitly to **-b nortllib** • **-b** symbolic is set only when building an executable or if not set explicitly by one of **-b symbolic**, **-b nosymbolic**, or **-b nosymbolic-** • **-b expfull** is set only when neither **-b E** nor **-b** export are present • **-b noexpall** is set • **-d**, instead of being ignored, is redefined and can assume two values: **-dy** or **-dn** • **-R**, instead of being ignored, takes one suboption that defines the runtime library search path • **-z**, instead of being a synonym of the **-K** option, takes either **defs**, **nodefs**, **multidefs**, **text**, **nowarntext**, or **warntext**, as a suboption • directories specified with the **-L** option are not included in the runtime libraries search path
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. This is the default when the **svr4** option is used; otherwise, the default is the **symbolic-** option.
textro	Same as the **ro** option.
textpsize:*psize*	Requests *psize* page sizes in bytes for text. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b textpsize:16k` or `-b textpsize:0x4000` will request 0x4000 for text and set the `F_VARPG` bit in the **XCOFF** header.
tmplrename	Specifies that the binder should check for general instantiations. The binder checks for any symbol of the form **__tfNNxxx**_*name* and renames the symbol to *name*. The default is **-bnotmplrename**. **Note:** This option is only needed for 32-bit mode. This option is ignored when building 64-bit objects.
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 *Files Reference*.
weaklocal	Specifies that weak symbols are searched for first in the object files where they are referenced. If the symbols are not found there, the normal search order is resumed.

Item	Description
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.

Symbol Visibility

Global and weak symbols in an input object file can be marked with a visibility. Four symbol visibilities are defined.

Table 3. Symbol Visibility

Symbol	Visibility
Internal	Symbol is not exported. The address of the symbol must not be provided to other programs or shared objects, but the linker does not verify this.
Hidden	Symbol is not exported
Protected	Symbol is exported but cannot be rebound (or preempted), even if runtime linking is being used.
Exported	Symbol is exported with the global export attribute.

The visibility of a symbol can be specified in an assembler source file. Some compilers support visibility as well. Consult your compiler documentation for details.

Export files can also be used to specify the visibility for a symbol. Ordinarily, the visibility specified in an export file takes precedence over the visibility specified in the object file. This linker considers symbol visibility when creating the export list for a program or shared object.

Import and export File Format (-bI: and -bE: Flags)

Each line within an import or export file contains the name of a symbol, optionally followed by an address or a keyword. Primary keywords are **svc**, **svc32**, **svc3264**, **svc64**, **syscall**, **syscall32**, **syscall3264**, **syscall64**, **symbolic**, **nosymbolic**, **nosymbolic-**, **list**, **cm**, **bss**, **internal**, **hidden**, **protected**, and **export**. Additional keywords are **weak** and **required**, which can be used in conjunction with another keyword.

In an import file, specifying an address allows a symbol to be mapped to a fixed address, such as an address in a shared memory segment. 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**, **svc32**, **svc3264**, **svc64**, **syscall**, **syscall32**, **syscall3264**, or **syscall64** keyword after a function name to indicate that the function is a system call. This is needed when linking kernel extensions. If the output file is not a kernel extension, these keywords are equivalent to the **symbolic** keyword.

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 symbols. 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 runtime linker.

You can use the **symbolic**, **nosymbolic**, or **nosymbolic**- keyword to associate an attribute with an exported symbol. A symbol address in an export file is ignored. In an export file, the keywords **cm** and **bss** are equivalent to the **nosymbolic** keyword. The visibility of a symbol can be specified with the **internal, hidden**, **protected**, or **export** keywords. For more information, see."Attributes of Exported Symbols" on page 208

The **weak** keyword can be used to specify weak symbol binding, and may be used with another attribute.

Use the **required** keyword to verify that a symbol is defined and not imported. An error is printed for symbols not meeting these criteria.

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

Item	Description
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:

Item	Description
#!	(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 *General Programming Concepts: Writing and Debugging Programs*. For more information on loading and binding, see the **load** subroutine in *Technical Reference: Base Operating System and Extensions, Volume 1*.

Attributes of Exported Symbols

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.

If a symbol is listed in an export file without a keyword, and the visibility of the symbol is specified in an input file, the symbol's visibility is preserved. An input symbol's visibility can be overridden by using the **internal, hidden**, **protected**, or **export** keyword.

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, or with an explicit visibility, the explicit attribute or visibility 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:

Item	Description
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 **-bI:/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

Item	Description
/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:

a.out file

load command

loadquery command

Shared Library Overview

Kernel Extensions and Device Support Programming Concepts

---

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

Item	Description
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

Item	Description
/usr/bin/ldd	Contains the **ldd** 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 displays a usage message using the standard error output.

Flags

Item **Description**

-b *Option* Modifies an executable as specified by *Option*. The possible values for *Option* are:

datapsize:*psize*
Requests *psize* page sizes in bytes for data. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix:

- k or K for kilo or 0x400 bytes
- m or M for mega or 0x100000 bytes
- g or G for giga or 0x40000000 bytes
- t or T for tera or 0x10000000000 bytes
- p or P for peta or 0x4000000000000 bytes
- x or X for exo or 0x1000000000000000 bytes

For example, either `-b datapsize:16k` or `-b datapsize:0x4000` will request 0x4000 for data and set the `F_VARPG` bit in the **XCOFF** header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the `F_VARPG` bit of the **XCOFF** header's **f_flags** member is also set. Otherwise, this bit is cleared.

forkpolicy:*policy*
Sets the _AOUT_FORK_POLICY flag in the XCOFF auxiliary header, if *File* is a 64-bit program. If *policy* is **cor**, the _AOUT_FORK_COW is also set, requesting the use of the copy-on-write forktree policy. If *policy* is **cow**, the _AOUT_FORK_COW flag is cleared, requesting the use of the copy-on-reference forktree policy. If *File* is a 32-bit program, no change is made.

noforkpolicy
Resets the _AOUT_FORK_POLICY and _AOUT_FORK_COR flags in the XCOFF auxiliary header, if *File* is a 64-bit program. The default forktree policy is used, unless a forktree policy is specified with the VMM_CNTRL environment variable. If *File* is a 32-bit program, no change is made.

lpdata Marks a file as a 'large page data' executable.

nolpdata
Unmarks a file as a 'large page data' executable.

noshrsymtab
Clears the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 32-bit object, no change is made.

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.

norwexec
Marks a file's writable and mappable sections and stack as non-executable.

rwexec Marks a file's writable and mappable sections and stack as executable.

shrsymtab
Sets the _AOUT_SHR_SYMTAB flag in the XCOFF auxiliary header. If *File* is a 64-bit program, a shared symbol table is created when the program is run. If *Flag* is a 64-bit object but not a 64-bit program, the _AOUT_SHR_SYMTAB flag can be set, but has no effect at runtime. If *File* is a 32-bit object, no change is made.

Item	Description
	stackpsize:*psize* Requests *psize* page sizes in bytes for process main thread stack. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b stackpsize:16k` or `-b stackpsize:0x4000` will request 0x4000 for process main thread stack and set the `F_VARPG` bit in the **XCOFF** header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the `F_VARPG` bit of the **XCOFF** header's **f_flags** member is also set. Otherwise, this bit is cleared.
	textpsize:*psize* Requests *psize* page sizes in bytes for text. The value can be specified as a decimal, hexadecimal, or octal number. The number specifications are the same as in C programming language. Additionally, the page sizes can be specified as a number followed by a one-character suffix: • k or K for kilo or 0x400 bytes • m or M for mega or 0x100000 bytes • g or G for giga or 0x40000000 bytes • t or T for tera or 0x10000000000 bytes • p or P for peta or 0x4000000000000 bytes • x or X for exo or 0x1000000000000000 bytes For example, either `-b textpsize:16k` or `-b textpsize:0x4000` will request 0x4000 for text and set the `F_VARPG` bit in the **XCOFF** header. It is accomplished by setting the corresponding member of the auxiliary header to the logarithm base 2 of the given value *psize*. If the value is different from 0, the `F_VARPG` bit of the **XCOFF** header's **f_flags** member is also set. Otherwise, this bit is cleared.
-V	Prints the version of the **ldedit** command on the standard error output.

Example

1. To request system-selected page sizes for text, data, and stacks, enter:
   ```
   ldedit -b textpsize:0 -b datapsize:0 -b stackpsize:0
   ```
   This clears the `F_VARPG` bit in the **XCOFF** header.

Related information:

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

Item	Description
-*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

Item	Description
/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:

ex command

Shells command

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

Item	Description
+	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:

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

Item	Description
Device_Name	The name of the LE Client device, for example, *ent1*.

Flags

Item	Description
-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 reference:

"mpcstat Command" on page 897

Related information:

entstat command

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.

Item	Description
%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).

User Subroutines

The **lex** library defines the following subroutines as macros that you can use in the rules section of the **lex** specification file:

Item	Description
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 character (Do not use the actual newline character in an expression.)
\r	Return
\t	Tab
\v	Vertical Tab
****	Backslash
\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, 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

Item	Description
-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:

Item	Description
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

Item	Description
/usr/ccs/lib/libl.a	Contains the run-time library.
/usr/ccs/lib/lex/ncform	Defines a finite state machine.

Related information:

yacc command

Generating a Lexical Analyzer with the lex Command

Using the lex Program with the yacc Program

National Language Support Overview for Programming

---

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 workstation.

Exit Status

This command returns the following exit values:

Item	Description
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.

Related information:

echo command

sh command

read command

Input and Output Handling Programmer's Overview

---

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

Item	Description
/usr/sbin/link	Contains the **link** command.

Related reference:

"ln Command" on page 239

Related information:

unlink command

link subroutine

fsck command

---

lint Command

Purpose

Checks C and C++ language programs for potential problems.

Syntax

lint [**-a**] [**-b**] [**-c**] [**-C**] [**-h**] [**-l***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:

Item	Description
/*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

Item	Description
-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.

Item	Description
-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):

Item	Description
-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:

Item	Description
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

Item	Description
/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 reference:

"make Command" on page 618

Related information:

cpp 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

Item	Description
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

Item	Description
/usr/bin/listdgrp	Contains the System V **listdgrp** command.

Related reference:

"lsattr Command" on page 354

"lsdev Command" on page 396

"mkdev Command" on page 701

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**] [**-n**] [**-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, USB disks, or tapes.

The **listvgbackup -r** and **restorevgfiles** commands perform identical operations and should be considered interchangeable.

Flags

Item	Description
-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 or technology 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 or technology 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**.
-n	Does not restore ACLs, PCLs, or extended attributes.
-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

Item	Description
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
   ```
10. To list the contents of the system backup located on device **/dev/usbms0,** use the following command:
    ```
    listvgbackup –f /dev/usbms0
    ```

Files

Item	Description
/usr/bin/listvgbackup	Contains the **listvgbackup** command

Related information:

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

Item	Description
ODM could not open class	The ODM database is not stored in the **/usr/lib/objrepos** directory.

Related information:
addX11input command

---

livedumpstart Command

Purpose

Initiates a live dump.

Syntax

livedumpstart [**-e**] [**-h**] [**-p** *pseudo-component*] [**-q**] [**-r**] [**-u**] [**-c** *component_path*] [**-l** *logical_alias*] [**-t** *type*] [**-C** *component_path* | **-L** *logical_alias* | **-T** *type*] *attribute* [...]

Description

The **livedumpstart** command is used to start a live dump. The dump can include one or more components. Only serialized dumps are used. It can be limited to one pass. The data acquired is dumped to the file system, and the dump is placed in a directory. The dump can be designated as informational or critical.

Components are dumped in the order that you specify. Specify the failing component with either the **-C**, **-L**, or **-T** flag. You cannot specify the name of a pseudo-component.

The data is dumped at the detail level that you set for that component, see the **dumpctrl** command for more information about managing system and live dumps.

If you do not specify the **-q** flag, the **livedumpstart** command displays a message containing the name of the dump.

Flags

Item	Description
-c *[+] component_path[+] [:parameter_list]*	Specifies a component by component path name. You can specify the **-c** flag more than once. If you precede a component name with a plus sign (+), the data from that component and its ancestors are dumped. If you follow a component name with a plus sign (+), the data from that component and its descendents are dumped. You can pass parameters to the component. Follow the component name and the optional "+" *:parameter_list*. A *parameter_list* consists of parameters separated by commas. It can also be groups separated by blanks. If a component and its ancestors or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
-C *[+] component_path[+] [:parameter_list]*	Specifies a failing component by component path name. At most one failing component can be specified. Thus, only one of the **-C**, **-L**, and **-T** flags is allowed, and that component specification must refer to a single component. If **-C basecomp+** is specified, and **basecomp** is not live dump aware, then only one component among **basecomp** and its descendents can be live dump aware. However, if **basecomp** is live dump aware, **basecomp** is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a component is preceded with a plus sign, "+", then that component and its ancestors are dumped. If a component is followed with a plus sign, "+", then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.

Item	Description
-e	Displays an estimate for the size of the dump, which contains the specified components or pseudo-components. This flag obtains a size estimate for the dump without starting the dump. To get an accurate estimate, use the same components, parameters, and detail level that you intend to use for the dump. The estimate takes into account the compression factor.
-h	Shows help text. If the **-h** flag is specified with other components or pseudo-components, the help text for those components is shown.
-l *[+] logical_alias[+] [:parameter_list]*	Specifies a component by component logical alias. You can specify multiple **-c**, **-l**, and **-t** flags. If a logical alias is preceded with a plus sign "+", then that alias and its ancestors are dumped. If a logical alias is followed with a plus sign "+", then that alias and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors, descendents, or both are specified, parameters are passed only to the component, not to the ancestors or descendents.
-L *[+] logical_alias[+] [:parameter_list]*	Specifies a failing component by component logical alias. At most one failing component can be specified. Thus, only one of the **-C**, **-L**, and **-T** flags is allowed, and that component specification must refer to a single component. If **-L basecomp+** is specified, and **basecomp** is not live dump aware, then only one component among **basecomp** and its descendents can be live dump aware. However, if **basecomp** is live dump aware, **basecomp** is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a logical alias is preceded with a plus sign "+", then that alias and its ancestors are dumped. If a logical alias is followed with a plus sign "+", then that alias and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
-p *pseudo-component [:parameter_list]*	Specifies a pseudo-component. **Note:** A pseudo-component (**-p**) cannot be a failing component. If parameters are to be passed to the pseudo-component, the pseudo-component must be followed by a *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. The following table is the description of pseudo-components.

Specification	Parameters	Description
eaddr:hex,hex	address and length, hexadecimal values	Dumps memory by kernel effective address.
context:addr=hex-eaddr | tid_t=hex-tid_t | cpu=dec-lcpu | bid=dec-bid	hex-eaddr - context (MST) effective address, hex-tid_t - kernel thread id, dec-lcpu - logical cpu, dec-bid - cpu bind id	Dumps a kernel context. This includes lightweight memory trace data, stack and thread state information.
tid_t:hex-tid	hexadecimal kernel thread id	Dumps a kernel thread by kernel thread ID.
tid:dec-tid	decimal kernel thread id	Dumps a kernel thread by kernel thread ID, and the ID is decimal.
tslot:dec-slot	decimal kernel thread slot number	Dumps a kernel thread by kernel thread ID, and the thread is specified by decimal slot number.
pid_t:hex-pid	hexadecimal kernel process id	Dumps a process by process ID.
pid:dec-pid	decimal process id	Dumps a process by process ID, and the ID is decimal.
pslot:dec-slot	decimal process slot number	Dumps a process by process ID, and the process is specified by decimal slot number.
errbuf	no parameters	Dumps kernel error logging data.
mtrc:common-size, rare-size	common and rare decimal buffer sizes	Dumps lightweight memory trace data.
systrace:dec-size	decimal buffer size	Dump system trace data. If the buffer size is 0, the entire buffer is dumped.
comptrace:component, dec-length	component name and decimal amount of data. The component can be an alias, and the length can be zero to dump the entire buffer.	Dumps component trace data.
kernext:pathname	extension's full path name	Allows symbol resolution for this extension.

-q	Specifies quiet mode. No messages are displayed.
-r	Dumps data for any subcomponents of the specified components. Specifying this flag is equivalent to specifying every component followed by a "+".
-t *[+] type[+] [:parameter_list]*	Specifies a component by its type or subtype. You can specify multiple **-c**, **-l**, and **-t** flags. If a type or subtype is preceded with a plus sign (+), then that component and its ancestors are dumped. If a type or subtype is followed with a plus sign (+), then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors and/or descendents are specified, parameters are passed only to the component, not to the ancestors or descendents.
-T *[+] type[+] [:parameter_list]*	Specifies a failing component by component type or subtype. At most one failing component can be specified. Thus, only one of the **-C**, **-L**, and **-T** flags is allowed, and that component specification must refer to a single component. If **-T type+** is specified, and *type* is not live dump aware, then only one component among *type* and its descendents can be live dump aware. However, if a component of the type *type* is live dump aware, it is the failing component, and it might have multiple live-dump-aware descendents. **Tip:** These rules also apply to a component and its ancestors. If a type or subtype is preceded with a plus sign (+), then that component and its ancestors are dumped. If a type is followed with a plus sign (+), then that component and its descendents are dumped. If parameters are passed to the component, the component and the optional "+" are followed with *:parameter_list*. A *parameter_list* consists of parameters separated by commas, or keyword=parm_list pairs separated by blanks. See the section on specifying parameters from the command line for more information. Note that if a component and its ancestors, descendents, or both are specified, parameters are passed only to the component, not to the ancestors or descendents.
-u	Dumps the data for the components that are "above" the specified components in the component hierarchy. This is equivalent to specifying every component preceded by a "+".

You can use wildcard when you specify component names and aliases. Remember that any parameters that you specify are passed to all matching components. You cannot use **all** or an asterisk (*).

Restriction: You can only specify one failing component, so **-C comp*** can resolve to only one component.

Attributes

The dump attributes are specified with keyword=value pairs. They are used to configure dump parameters, construct dump headers and edit symptom information. You change attributes by specifying an *Attribute=Value* parameter. If you have the proper authority you can set the following required attribute:

Item	Description
symptom=*string*	Provides symptom string details that must be supplied to further qualify the dump. The maximum length of this string is 2047 characters.

If you have the proper authority you can set the following optional group attributes:

Item	Description
errcode=*code*	Specifies the error code for the symptom string. If it begins with 0x, the value is in hex; if it begins with 0, the value is octal; otherwise it is decimal.
force=yes | no	If yes, overrides duplicate checking, dumps the data regardless of whether it duplicates a previous dump. The default is **yes**, because any dump taken from the command line should not be treated as a duplicate.
log=yes | no	Specifies whether a log entry should be written when the dump completes. If you specify **yes**, a message is written to the error log. The default is **yes**.
noforce	For software initiated live dumps. Specifies whether to initiate the dump if it duplicates a previous dump that was initiated within the last day. The **noforce** attribute makes the dump subject to duplicate elimination.
nolog	Specifies whether to a message should be written to the error log when the dump is complete. If it is not specified, dump completion and errors are logged.
prefix=*prefix*	Specifies the file name prefix. The name can be no more than 63 characters.
priority=*priority*	Specifies the priority of the dump. You can specify **info** or **critical**. The default is **critical**. If you specify the value **info**, it indicates the dump is for informational purposes, while **critical** indicates the dump is necessary to debug a problem.
title=*string*	Specifies an optional dump title, which can be up to 127 characters.
type=*type*	Specifies whether data should be collected without freezing the system. **serialized | ser** The dump data is gathered when the system is frozen. It might be necessary to use multiple freezes to dump all the data. This is the default. **unserialized | unser** Data is gathered without freezing the system. It might be necessary to use multiple freezes to dump all the data. If you specify **unserialized**, the system is not frozen when gathering the data.
onepass	All data is gathered under one pass. The dump is truncated if all data does not fit in available memory. The default is to use multiple passes if required.

Exit Status

Item	Description
0 (zero)	The **livedumpstart** command completes successfully and produces a message containing the name of the dump.

Item	Description
nonzero	The **livedumpstart** command fails and produces an error message. This command fails under the following conditions: • One or more parameters are not valid. • One or more components are not valid. • None of these components can be specified for a live dump. • A component attempts to take a live dump from within a live dump. • Live dumps are disabled. • A dump already exists. This can occur when you specify the **force=no** attribute. • There is insufficient memory. • All the data cannot be buffered in this single-pass dump. • Too much time is spent while processors are disabled, and this dump is truncated.

Security

Only the root user can run this command.

Examples

1. To dump data for device `ent0`, and components above it in the component hierarchy, enter the following command:

   ```
   livedumpstart -L +ent0 symptom=foo
   ```

   The failing component is `ent0`. This creates a dump named `ent0.yymmddhhmm.00.DZ`. It is a serialized, critical dump.

   Tip: According to the rules for specifying the failing component, if `ent0` is not live dump aware, but multiple ancestors are, then this command fails. If `ent0` is not live dump aware, and only one ancestor is, this ancestor is used as the failing component.
2. To create an informational dump of the process management data for processes `856` and `10272`, enter the following command:

   ```
   livedumpstart -p pid:856 -p pid:10272 \
                 info prefix=mydump title="process dump" symptom="foo"
   ```

   The dump is named `mydump.nocomp.yymmddhhmm.00.DZ`. Note there is no failing component.
3. To create a serialized, one-pass dump where `foo` is the failing component, enter the following command:

   ```
   livedumpstart -C foo+:block=45ab8 -pcontext:tid_t=57B29 onepass symptom=bar
   ```

   This command dumps `foo`, its descendents, and the context for kernel thread `57B29`. The dump is named `foo.yymmddhhmm.00.DZ`.
4. A subsystem has the parent component with alias `subsyst`. It has only one live-dump-aware component. To create a serialized live dump of this subsystem, you might use the following command:

   ```
   livedumpstart -L subsyst+ title="Dump of subsystem subsyst" symptom=foo
   ```
5. To specify that process `1234` is dumped along with 0x400 bytes starting at 0x45928, enter the following command:

   ```
   livedumpstart -p tid:1234 -p eadder:45928,400 symptom=foo
   ```

   In this example, there is no failing component.

lkdev Command

Purpose

Locks a device. Any attempt to modify device characteristics fails.

Syntax

lkdev [**-l** *Name* **-a** | **-d** [**-c** *Text*]]

lkdev -h

Description

The **lkdev** command locks the specified device (the **-l** *Name* flag). Any attempt to modify device attributes by using the **chdev** or **chpath** command is denied. In addition, an attempt to delete the specified device or one of its paths from the Object Data Manager (ODM) by using either the **rmdev** or **rmpath** command is denied.

Flags

Item	Description
-h	Displays the command usage message.
-l *Name*	Specifies the logical device name of the target device for the paths affected by the change. This flag is required in all cases.
-a	Locks the specified device.
-d	Unlocks the specified device.
-c *Text*	Specifies a text string of up to 64 printable characters that contain no embedded spaces.

Security

Privilege control: Only the root user can execute this command.

Auditing events:

Event	Information
DEV_LOCK	The device command line

Examples

1. To enable the lock for the *hdisk1* disk device, enter the following command:
   ```
   lkdev -l hdisk1 -a
   ```
2. To disable the lock for the *hdisk1* disk device, enter the following command:
   ```
   lkdev -l hdisk1 –d
   ```
3. To enable the lock for the *hdisk1* disk device and create a text label, enter the following command:
   ```
   lkdev -l hdisk1 -a –c test_string
   ```
4. To modify the text label for the *hdisk1* disk device, enter the following command:
   ```
   lkdev -l hdisk1 –c new_test_string
   ```

Location

Item	Description
/usr/sbin/lkdev	Contains the **lkdev** command

Related information:
chdev command
chpath command
rmdev command
rmpath command

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

Item	Description
-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 behavior.
-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:

Item	Description
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

Item	Description
/usr/bin/ln	Contains the **ln** command.

Related information:

rm command

link command

symlink command

Linking for Programmers

---

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

Item	Description
-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:

Item	Description
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 (").

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 reference:

"localedef Command"

Related information:

Character Set Description (charmap) Source File Format

Locale Definition Source File Format

Understanding Locale Environment Variables

---

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

Item	Description
-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:

Item	Description
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 reference:

"locale Command" on page 241

"ld Command" on page 190

Related information:

setlocale command

Method Source File Format

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

Item	Description
-*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

Item	Description
/usr/bin/lock	Contains the **lock** command.

Related information:

passwd command

passwd

lockd Daemon

Purpose

Processes lock requests.

Syntax

/usr/sbin/rpc.lockd [**-t** *TimeOut*] [**-g** *GracePeriod*] [**-d** *debug*] [**-x** *xnfs*] [**-T** *RetransmissionsTimeout*] [*number of server*]

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 about the location of connections as well as the status in the **/var/statmon/sm** directory, the **/var/statmon/sm.bak** file, and the **/var/statmon/state** file. When restarted, the **statd** daemon queries these files and tries to reestablish the connection it had prior to termination. To restart the **statd** daemon, and subsequently the **lockd** daemon, without prior knowledge of existing locks or status, delete these files before restarting the **statd** daemon.

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

Item	Description
-d *debug*	Specifies the debug level of the **rpc.statd** daemon. By default, the debug level is disabled.
-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 *RetransmissionsTimeout*	Specifies the timeout for one-way RPC connections. One-way RPC connections stay valid for *RetransmissionsTimeout* number of seconds. If this variable is set to 0, then there is no client cache for one-way RPC calls. The default value for the *RetransmissionsTimeout* variable is 300 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.
-x *xnfs*	Specifies if the **rpc.lockd** daemon needs to follow the xnfs specification. By default, this flag is turned off.

Parameters

Item	Description
number of server	Specifies the number of daemons to start.

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

Item	Description
/etc/services	Contains **lockd** parameter information entries.

Related information:

no command

lockf command

signal command

Network File System (NFS) Overview for System Management

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

Item	Description
-r *classname*	Turns off lock tracing for all kernel locks belonging to the specified class. The option always fails if you did not run the `bosboot -L` command.
-s *classname*	Turns on lock tracing for all kernel locks belonging to the specified class. The option always fails if you did not run the `bosboot -L` command. To trace several specific classes at the same time, run the **locktrace** command multiple times, with a specific lock class each time. You can enter up to 32 class names.
-R	Turns off all lock tracing.
-S	Turns on lock tracing for all locks regardless of their class membership.
-l	Lists kernel lock tracing current status.

Examples

1. To start tracing the SEM_LOCK_CLASS, enter the following command:

   ```
   locktrace -s SEM_LOCK_CLASS
   ```

2. To stop all lock tracing, enter the following command:

   ```
   locktrace -R
   ```

3. To reset previous lock trace entries and then trace the SEM_LOCK_CLASS and SHM_LOCK_CLASS lock classes, enter the following commands:

   ```
   locktrace -R
   locktrace -s SEM_LOCK_CLASS
   locktrace -s SHM_LOCK_CLASS
   ```

   You can view current lock classes using the **-l** flag:

   ```
   locktrace -l
   ```

   The following output will be displayed:

   ```
   lock tracing enabled for classes:
     SHM_LOCK_CLASS
     SEM_LOCK_CLASS
   ```

File

Item	Description
/usr/bin/locktrace	Contains the **locktrace** command.
/usr/include/sys/lockname.h	Contains the **lock** class names.

Related information:

bosboot command

trace command

logevent Command

Purpose

Logs event information generated by the event response resource manager (ERRM) to a specified log file.

Syntax

logevent [**-h**] *log_file*

Description

The **logevent** captures event information that is posted by the event response resource manager (ERRM) in environment variables the ERRM generates when an event occurs. This script can be used as an action

that is run by an event response resource. It can also be used as a template to create other user-defined actions. The language in which the messages of the **logevent** script are returned depend on the locale settings.

Event information that is returned about the ERRM environment variables 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.

This script uses 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

- This script must be run on the node where the ERRM is running.
- The user who runs this script 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** as follows:

   ```
   /opt/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 2007
   ```

```
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

/opt/rsct/bin/logevent

---

logform Command

Purpose

Initializes a logical volume for use as a Journaled File System (JFS) log. Initializes an Enhanced Journaled File System (JFS2) outline log. Reformats an inline log for an existing JFS2 file system using an inline log.

Syntax

logform [**-V** *vfstype*] *LogName*

Description

The **logform** command initializes a logical volume for use as a JFS or JFS2 log device. Running the **logform** command on any JFS log device or JFS2 outline or inline log device will destroy all log records on the log device. This may cause the file system to lose its recovery capability and therefore to lose the file system data.

When you run the **logform** command on an outline log for a JFS2 file system that is already using an outline log, the device type for the outline log must be **jfs2log**. Otherwise, the **logform** command will exit with an error.

To reuse an existing logical volume as an outline log device for a JFS2 file system, you must delete the logical volume and then recreate it as device type **jfs2log**.

For the outline log device of a JFS file system, the same rules are applied. That is, for a new logical volume, the type should be `jfslog`. For a reuse logical volume, you should delete the logical volume and recreate it as lv type `jfslog`. However, logform does not do type check for the log device of a JFS file system. The **logform** command does not report any error when input log device has a wrong lv type. The user should pay attention to the lv type.

When you run the **logform** command on a device with logical volume type **jfs2**, if the device has a file system with an inline log, then the inline log will be reformatted. If the device has a file system with an outline log, then an error will be reported.

When the **logform** command is used to format an inline log for an existing JFS2 file system, the file system data will not be affected; only the log records are destroyed. The logical volume type for an inline log is the same as for the file system. For a JFS2 file system, the inline log logical volume type is **jfs2**.

For a JFS2 file system, **logform** formats a maximum of 2047 MBytes of log. If the log size is greater than 2047 MBytes, only 2047 MBytes will be formatted and the rest will left untouched and will not be used.

Flags

Item	Description
-V *vfstype* [jfs | jfs2]	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. Note that for jfs2 log device this flag is always ignored. The **logform** command is unable to change the lv type according to the value of the **-V** flag. Therefore the user should create an lv with the correct lv type (jfslog or jfs2log) before calling the **logform** command. Use of this flag is strongly discouraged.

Parameters

Item	Description
LogName	The *LogName* parameter specifies the absolute path to the logical volume to be initialized (for example, /dev/jfslog1). When the **logform** command is run on an inline log, *LogName* is the device name of the file system.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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.
3. To format the inline log for an existing file system called /j2 which is on the file system device /dev/fslv00, type:

   ```
   logform /dev/fslv00
   ```

   This formats the inline log for file system /j2, but does not touch the data in the file system.

Files

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics, including the log device.

Related reference:

"mkfs Command" on page 716

"mklv Command" on page 739

Related information:

crfs command

File systems

Mounting command

JFS and JFS2

logger Command

Purpose

Makes entries in the system log.

Syntax

logger [**-f** *File*] [**-i**] [**-r** *[Count]*] [**-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

Item	Description
-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.
-r *Count*	If the buffer resource is not available, retries logging the message for the specified number of times. If the number is not specified, retries logging the message until the message is logged. The number must be a positive integer ranging from 1 - 1000.

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:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/bin/logger	Contains the **logger** command.

Related information:

syslogd command

syslog command

---

login Command

Purpose

Initiates a user session.

Syntax

login [**-h** *HostName*] [**-p**] [**-f** *User* | **-k**] [**-e** *Label*] [**-t** *Label*] [*User* [*Environment*]]

Description

The **login** command (part of the **tsm** command) initiates sessions on the system for the user that is specified by the *User* parameter. You can also specify environment variables to be added to the user's environment. These commands are strings of the form *Variable*=*Value*. The **login** command is not normally entered on the command line.

You can configure the **login** command to create your home directory at your login if you do not have a home directory already. The **login** command calls the **mkuser.sys** command to create the home directory and customize the account. To enable this capability, set the **mkhomeatlogin** attribute of the **usw** stanza in the **/etc/security/login.cfg** file to `true`.

Note:

1. The **PATH**, **IFS**, **HOME**, and **SHELL** environment variables can not be initialized from the command line.
2. The **login** command supports multibyte user names. It is recommended that the system administrator must restrict the user names to characters within the portable character set, to avoid 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 restart the system.
4. If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, all group IDs are fetched from the LDAP module, and from the files modules, if the user belongs to any one of these domains.

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:

Item	Description
Checks accounts	The **login** command validates the user's account, ensuring authentication, logins enabled properly, and correct capacity for the port that is 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 logging 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 that is stored is the same as that in the **/etc/utmp** file, except that unrecognizable user names are logged as `UNKNOWN_USER`. This check 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.

To preserve the integrity of the system, only one session at a time is allowed to be logged in to a port. This check 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.

On a Trusted AIX system, you can specify an effective sensitivity label (SL) at login time by specifying the label with the **-e** flag along with the user name. To specify an effective integrity label (TL) during login, specify the label by using the **-t** flag.

If the label has spaces, specify it within quotation marks. The default login SL and TL are defined in the **/etc/security/user** file as user attributes. If no label attribute is specified in the file, the label attributes that are defined in the default stanza are used.

The labels that you supply must be dominated by your clearance and contained in the system accreditation range. You can specify the SL with the **-e** flag and the TL with the **-t** flag at login time. In a labeled network, unless the login is done by using the console, the network's label is assigned to you, regardless of the labels that you specified with the **-e** or **-t** flag.

Your SL clearance must be within the range that is defined for the TTY device in the **/etc/security/login.cfg** file. The effective TL of the user must be the same as the TL of the TTY. After successfully logging in, the clearance is assigned to the login port.

Tip:

Unless your terminal displays only uppercase letters, do not use only uppercase characters for your user name.

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

Item	Description
-e *Label*	Specifies the effective sensitivity label to be used to log in to a Trusted AIX system. **Restriction:** The **-e** flag applies only to systems that are running Trusted AIX.
-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 system that is requesting the login. This form of the login is used only by the **telnetd** and **rlogind** daemons.
-k	Identifies the login as using Kerberos authentication and causes login to pass control to **/usr/bin/k5dcelogin** to handle authentication. This form of login is only used by the **krshd** daemon.
-p	Preserves the current terminal type by setting it the value of the **$TERM** environment variable instead of the type that is contained in the **CuAt/PdAt** object classes database.
-t *Label*	Specifies the effective integrity label to be used to log in to a Trusted AIX system. **Restriction:** The **-t** flag applies only to systems that are running Trusted AIX.

Security

The **login** command is a PAM-enabled application with a service name of `login`. System-wide configuration to use PAM for authentication is set by modifying the value of the **auth_type** attribute, in the `usw` stanza of **/etc/security/login.cfg**, to `PAM_AUTH` as the root user.

The authentication mechanisms that are used when PAM is enabled depend on the configuration for the login service in **/etc/pam.conf**. The **login** command requires **/etc/pam.conf** entries for the `auth`, `account`, `password`, and `session` module types. The following is a recommended configuration in **/etc/pam.conf** for the login service:

```
#
# AIX login configuration
#
login auth required /usr/lib/security/pam_aix

login account required /usr/lib/security/pam_aix

login session required /usr/lib/security/pam_aix

login password required /usr/lib/security/pam_aix
```

Examples

1. 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.
2. On a Trusted AIX system, to log in to the system as user `james`, with the effective SL of `TOP SECRET`, enter the following command:

   ```
   login: james –e "TOP SECRET"
   ```
3. To log in with the effective SL of `SECRET`, and the effective TL of `TOP SECRET`, enter the following command:

   ```
   login: james –e "TOP SECRET" –t "TOP SECRET"
   ```
4. On the command line the following can be used:

```
$ login –e "TOP SECRET" james
```

Files

Item	Description
/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 that pertains to the most recent successful and unsuccessful login attempts.
/etc/security/failedlogin	Contains information that pertains to each unsuccessful login.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.

Related information:

getty command

lastlog command

authenticate command

Login message suppression

logins Command

Note: Logins command displays system login information details only for the local users or groups which are defined in the **/etc/passwd** and **/etc/group** files.

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 items:

- User 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

Item	Description
-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

Item	Description
/usr/bin/logins	Contains the **logins** command.
/etc/passwd	Contains the password file.
/etc/group	Contains the group file.

Related reference:
"lsuser Command" on page 550
"lsgroup Command" on page 412

---

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.

The **logname** command invokes the **getlogin** subroutine to get the information about the login name.

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:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

To display your login name to standard output, enter:

```
logname
```

Files

Item	Description
/usr/bin/logname	Contains the **logname** command.

Related information:

getty command

su command

tsm command

getlogin 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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

From a shell started by the **ksh** or **bsh** command, enter:

```
logout
```

Files

Item	Description
/usr/bin/logout	Contains the **logout** command.
/etc/utmp	Contains a record of logged-in users.

Related reference:

"login Command" on page 253

Related information:

getty command

shell command

tsm command

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

Item	Description
-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

Item	Description
/usr/share/dict/words	Contains the default dictionary.

Related information:

grep command

sort command

Input and output redirection

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

Item	Description
-n	Turns off the prompt for instructions.

Files

Item	Description
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 reference:

"indxbib Command" on page 42

Related information:

addbib command

roffbib command

sortbib command

loopmount Command

Purpose

Associate an image file to a loopback device. Optionally, make an image file available as a file system via the loopback device.

Syntax

loopmount { -i *imagefile* | -l *device* } [-o *mount options* -m *mountpoint*]

Description

This command is similar to **mount** except that it creates a loopback device if not specified, binds the specified file to it, and optionally mounts it. If the command implicitly creates a new loopback device, it sets the temporary attribute in CuAt to **yes** so that it will be deleted by subsequent loopumount or reboot. All the restrictions and features of the **mount** command apply to **loopmount** also.

Flags

Item	Description
-i	Specify an image file name such as an ISO image. This must be specified if –l is not specified.
-l	**ODM name** of a loopback device such as loop0, loop1, etc. This must be specified if –i is not specified.
-o	Options for the mount command.
-m	Mount point such as **/mnt**.

If -l and -i are both specified, the *imagefile* is associated with the device before mounting it.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following mount an ISO image on /mnt.

   ```
   $ loopmount -i cdrom.iso -o "-V cdrfs -o ro" -m /mnt
   ```

2. The following mounts a disk image on /mydisk with loop2. The image file was bound to loop2 earlier with **chdev** command.

   ```
   $ loopmount -l loop2 -o "-V jfs2 -o rw,log=NULL" -m /mydisk
   ```

   In case the filesystem was created with an INLINE log, this INLINE log can be used.

   ```
   $ loopmount -l loop2 -o "-V jfs2 -o rw,log=INLINE" -m /mydisk
   ```

3. The following mounts an image file bound to loop0 on /mnt.

   ```
   $ loopmount -i mycd.iso -l loop0 -o "-V cdrfs -o ro" -m /mnt
   ```

Files

Item	Description
/usr/sbin/loopmount	Contains the loopmount command.

Related information:

loopumount command

mount command

umount command

loopumount Command

Purpose

Unmounts a previously mounted image file on a loopback device and then removes the device.

Syntax

loopumount { -i *imagefile* | -l *device* } [-o *umount options* -m *mountpoint*]

Description

This command is similar to **umount** except that it unmount the file and deletes the loopback device associated with the mount point if the temporary attribute for the loopback device in CuAt is set to **yes**, and then unmounts it. All the restrictions and features of the **umount** command apply to **loopumount** also.

Flags

Item	Description
-i	Specify an image file name such as an ISO image. This must be specified if –l is not specified.
-l	**ODM name** of a loopback device such as loop0, loop1, etc. This must be specified if –i is not specified.
-o	Options for the **umount** command.
-m	Mount point such as /mnt.

If -i and –l both are specified, -i is ignored.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. The following unmount /mnt and deletes the underlying device that was created by a previous invocation of **loopmount**.

   ```
   $ loopumount –i cdrom.iso –o "/mnt"
   ```

2. The following command unmount /dev/loop2 from /mydisk but does not delete the device as loop2 was created by the user with **mkdev**.

   ```
   $ loopumount –l loop2 –o "/mydisk"
   ```

Files

Item	Description
/usr/sbin/loopumount	Contains the loopumount command.

Related information:

loopmount command

mount command

umount 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

Item	Description
-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

Item	Description
/tmp/sym*	Contains temporary files.

Related information:

ar command

tsort command

xargs command

ar command

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.

Flags

Item	Description
-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.
-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*	Specifies that flags specific to the backend be passed to the backend. Thus, for each queue, other flags not described in this article can be included with the **lp** command. See the **piobe** command for a list of these flags. Specifying this flag is the same as specifying the **-o** flag for the **enq** command.
-s	Suppresses the automatic return of job numbers. The **lp** command reports the job number as the default, the **-s** flag overrides the default.
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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:

Item	Description
0	All input files processed successfully.
>0	No output device is available, or an error occurred.

Files

Item	Description
/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.

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).

If you enter `lp -?`, the system displays the command usage message and returns 0.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

/var/spool/lp/*

Related reference:

"lpr Command" on page 323

"lpstat Command" on page 331

"mail, Mail, or mailx Command" on page 605

Related information:

enable command

/etc/qconfig file

Printing administration

---

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*

/usr/lib/lp/bin/lp.tell *Printer*

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 a newline character 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

Item	Description
-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**:

Item	Description
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:

Item	Description
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

Item	Description
/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

lpacl Information

Purpose

Provides general information about protecting the least-privilege (LP) commands resource class and its resources by using access controls that are provided by the resource monitoring and control (RMC) subsystem.

Description

RMC controls access to all of its resources and resource classes through access control lists (ACLs), using two different ACL implementations. The implementation that RMC uses depends on which class is involved. The two major differences between the implementations are in: 1) the mechanisms with which ACLs are viewed and modified and 2) whether ACLs are associated with individual resources.

RMC implements access controls for its resources and resource classes in the following ways:

1. Through ACLs that are defined by resource class stanzas in the **ctrmc.acls** file.

   You can view these ACLs by examining the **ctrmc.acls** file. You can modify these ACLs using the **chrmcacl** command. Use a stanza to define an ACL that applies to a class or to all of the resources in a class.

   RMC uses this method for all of its resources and resource classes, except for the **IBM.LPCommands** resource class and its resources.
2. Through ACLs that are associated with resources and a resource class within the RMC subsystem.

   You can view and modify these ACLs using LP commands. You can define an ACL that applies to a class or an ACL that applies to an individual resource of a class.

   RMC uses this method for the **IBM.LPCommands** resource class and its resources.

   This section provides information about ACLs that are specific to the **IBM.LPCommands** resource class and its resources.

The LP resource manager uses the **IBM.LPCommands** resource class to define LP resources. These resources represent commands or scripts that require **root** authority to run, but typically the users who need to run these commands do not have **root** authority. By using the LP resource manager commands, users can run commands that require **root** authority. The LP resource manager commands are:

chlpcmd
: Changes the read or write attribute values of an LP resource

lphistory
: Lists or clears a certain number of LP commands that were previously issued during the current RMC session.

lslpcmd
: Lists information about the LP resources on one or more nodes in a domain.

mklpcmd
: Defines a new LP resource to RMC and specifies user permissions.

rmlpcmd
: Removes one or more LP resources from the RMC subsystem.

runlpcmd
Runs an LP resource.

For descriptions of these commands, see Least-privilege (LP) resource manager commands in *Technical Reference: RSCT for AIX* for AIX and Least-privilege (LP) resource manager commands in *Technical Reference: RSCT for Multiplatforms* for other operating systems. For information about how to use these commands, see the *Administering RSCT* guide.

Because each LP resource can define a unique command, RMC implements ACLs for the **IBM.LPCommands** class that allows access to be controlled at the individual resource level and at the class level. RSCT provides a set of commands that you can use to list and modify the ACLs for the **IBM.LPCommands** class and its resources. The LP ACL commands are:

chlpclacl
Changes the Class ACL

chlpracl
Changes the Resource ACL

chlpriacl
Changes the Resource Initial ACL

chlprsacl
Changes the Resource Shared ACL

lslpclacl
Lists the Class ACL

lslpracl
Lists the Resource ACL

lslpriacl
Lists the Resource Initial ACL

lslprsacl
Lists the Resource Shared ACL

mklpcmd
Defines a new LP resource to RMC and specifies user permissions

Security

- To use the LP commands that change the Class ACL, the Resource Initial ACL, and the Resource Shared ACL, you must have query and administrator permission for the **IBM.LPCommands** class.
- To use the LP command that changes a Resource ACL for an LP resource, you must have query and administrator permission for the LP resource.
- To use the LP commands that list the Class ACL, the Resource Initial ACL, and the Resource Shared ACL, you must have query permission for the **IBM.LPCommands** class.
- To use the LP command that lists a Resource ACL for an LP resource, you must have query permission for the LP resource.

The **Security** section of each LP command description indicates which permissions are required for the command to run properly.

Implementation specifics

This information is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Location

/opt/rsct/man/lpacl.7

Examples

Some examples of how to modify the LP ACLs follow. In these examples, the commands are run on a management server for a group of nodes in a management domain. The management server is named **ms_node** and the managed nodes are called **mc_node1**, **mc_node2**, and so on. In a management domain, it is most likely that the LP resources are defined on the management server and the LP commands themselves are targeted to the managed nodes. In these examples, the Resource Shared ACL is not used because separate permissions are required for the individual LP resources. These examples assume that the LP resources are not yet defined by using the **mklpcmd** command.

1. You want to define the **lpadmin** ID to be the administrator for the LP commands. This ID has the authority to modify the LP ACLs. You also want to give this ID read and write permission to be able to create, delete, and modify the LP resources. To configure this setting, use the **root** mapped identity to run these commands on the management server:

   ```
   chlpclacl lpadmin@LOCALHOST rwa

   chlpriacl lpadmin@LOCALHOST rwa
   ```

   These commands define the **lpadmin** ID on the management server as having administrator, read, and write permission for the **IBM.LPCommands** class and for the Resource Initial ACL. The Resource Initial ACL is used to initialize a Resource ACL when an LP resource is created. Therefore, when an LP resource is created, the **lpadmin** ID has administrator, read, and write permission to it.
2. The **lpadmin** ID can now create LP resources that define the LP commands that are needed. Access to the LP resources can be defined using the **mklpcmd** command or the **chlpracl** command. When the resource is created, the Resource Initial ACL is copied to the Resource ACL. To modify the Resource ACL using the **chlpracl** command so that **joe** is able to use the **runlpcmd** command for the resource named **SysCmd1**, the **lpadmin** ID runs this command on the management server:

   ```
   chlpracl SysCmd1 joe@LOCALHOST x
   ```

   This command gives **joe** run permission on the management server to the **SysCmd1** resource so he can use the **runlpcmd** command.
3. In this example, only the **lpadmin** ID has permission to create, delete, and modify LP resources. Use the **chlpclacl** command so that other users can create and delete LP resources. In this case, they need to have write access to the class. To be able to list the resources in the **IBM.LPCommands** class, read permission is required. Read permission on a Resource ACL allows a user to view that LP resource. Write permission on a Resource ACL allows a user to modify that LP resource. To allow **joe** to view the LP resource named **SysCmd1**, the **lpadmin** ID runs this command on the management server:

   ```
   chlpracl SysCmd1 joe@LOCALHOST r
   ```
4. There are several nodes in a peer domain. There is an LP resource called **SysCmdB1** on **nodeB** for which **joe** needs run permission. In addition, **joe** needs to have run permission from nodes **nodeA**, **nodeB**, and **nodeD**. If you run the **chlpracl** command on **nodeB**, you can use **joe@LOCALHOST** for **nodeB**, but you need to determine the node IDs for **nodeA** and **nodeD**. To obtain the node IDs, enter:

   ```
   lsrpnode -i
   ```

   The following output is displayed:

   ```
   Name    OpState RSCTVersion NodeNum NodeID

   nodeA   Online  3.1.0.0     2       48ce221932ae0062

   nodeB   Online  3.1.0.0     1       7283cb8de374d123

   nodeC   Online  3.1.0.0     4       b3eda8374bc839de

   nodeD   Online  3.1.0.0     5       374bdcbe384ed38a

   nodeE   Online  3.1.0.0     2       ba74503cea374110
   ```

```
nodeF   Online  3.1.0.0     1       4859dfbd44023e13

nodeG   Online  3.1.0.0     4       68463748bcc7e773
```

Then, to give **joe** the permissions as stated earlier, run on nodeB:

```
chlpracl SysCmd1 -l joe@LOCALHOST joe@0x48ce221932ae0062 \

joe@0x374bdcbe384ed38a x
```

Related information:

lphistory command

lslpclacl command

runlpcmd command

ctsec_map.global 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**] [**-I** *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**] [**-I** *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*]

Setting or Changing the Printer's High Sensitivity Labels and Low Sensitivity Labels with Trusted AIX

lpadmin -p *Printer* **-J** *label* **-L** *label*

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 file system 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.

If you enter `lpadmin -?`, the system displays the command usage message and returns 0.

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

Item

-A *AlertType* **[-W** *minutes* **]**

Description

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 using 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.

Item	Description
	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 allow: *FormList* | **-f deny:***FormList*	Allows or denies the forms in *FormLlist* 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.
-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**.
-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 *Interface*	Establish a new interface program for *Printer*. The *Interface* is the path name of the new program. Do not specify the **-e** and **-m** flags with this flag.

Item	Description

-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.

-J *label*	Defines the high Sensitivity Label (SL) for a printer with Trusted AIX installed.
-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.
-L *label*	Defines the low Sensitivity Label (SL) for a printer with Trusted AIX installed.
-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**.

Item	Description
-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.

Item	Description
-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 *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. *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.
-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.
-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*	

Item	Description
-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 path name 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.

Notes:

- When creating a new printer, you must specify the **-v**, or **-s** flag. In addition, only one of the following can be supplied: **-e**, **-i**, or **-m**; if none of these three flags is supplied, the model standard is used.
- If you specify the **-s** or **-R** flags, the following flags are not valid: **-A**, **-e**, **-F**, **-h**, **-i**, **-l**, **-M**, **-m**, **-o**, **-v**, and **-W**.
- If you specify the **-J** or **-L** flag, you must specify both flags. The **-p** flag is the only other flag that you can specify with these two flags.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

/var/spool/lp/*

/etc/lp

Related reference:

“lpsched Command” on page 330

“lpsystem Command” on page 335

Related information:

accept command

lpar_netboot Command

Purpose

Retrieves the media access control (MAC) address and physical location code from network adapters for a partition, or instructs a partition to start the network.

Syntax

To retrieve the MAC address and physical location code:

lpar_netboot -M -n [**-v**] [**-x**] [**-f**] [**-i**] [**-E** *environment* [**-E ...**]] [**-A**] **-t ent** [**-T** *{ on | off }*] [**-D -s** *Speed* **-d** *Duplex* **-S** *Server* **-G** *Gateway* **-C** *Client* [**-K** *subnetmask*] [**-V** *vlan_tag*] [**-Y** *vlan_priority*]] *partition_name partition_profile managed_system*

To perform a network boot:

lpar_netboot [**-v**] [**-x**] [**-f**] [**-i**] [**-E** *environment* [**-E ...**]] [**-g** *args*] [{ **-A -D** | [**-D**] **-l** *physical_location_code* | [**-D**] **-m** *MAC-address* }] **-t ent** [**-T** *{ on | off }*] **-s** *Speed* **-d** *Duplex* **-S** *Server* **-G** *Gateway* **-C** *Client* [**-K** *subnetmask*] [**-V** *vlan_tag*] [**-Y** *vlan_priority*] [[**-a -B** *tftp_image_filename*] **-B** *bootp_image_filename*] *partition_name partition_profile managed_system*

To retrieve the MAC address and physical location code on a system supporting a full system partition:

lpar_netboot -M -n [**-v**] [**-x**] [**-f**] [**-i**] [**-E** *environment* [**-E ...**]] [**-A**] **-t ent** [**-T** *{ on | off }*] [**-D -s** *Speed* **-d** *Duplex* **-S** *Server* **-G** *Gateway* **-C** *Client* [**-K** *subnetmask*] [**-V** *vlan_tag*] [**-Y** *vlan_priority*]] *managed_system managed_system*

To perform network boot on a system supporting a full system partition:

lpar_netboot [**-v**] [**-x**] [**-f**] [**-i**] [**-E** *environment* [**-E ...**]] [**-g** *args*] [{ **-A -D** | [**-D**] **-l** *physical_location_code* | [**-D**] **-m** *MAC-address* }] **-t ent** [**-T** *{ on | off }*] **-s** *Speed* **-d** *Duplex* **-S** *Server* **-G** *Gateway* **-C** *Client* [**-K** *subnetmask*] [**-V** *vlan_tag*] [**-Y** *vlan_priority*] [[**-a -B** *tftp_image_filename*] **-B** *bootp_image_filename*] *managed_system managed_system*

Description

The **lpar_netboot** command instructs a logical partition to network boot, by having the partition send out a **bootp** request to a server specified with the **-S** flag. The server can be a NIM server that serves SPOT resources, or it can be another server that serves network boot images.

If the **-M** and **-n** flags are specified, the **lpar_netboot** command returns the Media Access Control (MAC) address and the physical location code for a particular type of network adapter that is specified with the **-t** flag. When the **-m** flag is specified, **lpar_netboot** boots a partition, by using a specific network adapter that matches the specified MAC address. When the **-l** flag is specified, **lpar_netboot** boots a partition, by using a specific physical location code for the network adapter that matches the specified physical location code. The matching MAC address or physical location code is dependent upon the hardware resource allocation in the profile in which the partition was booted. The **lpar_netboot** command also requires arguments for partition name, partition profile (which contains the allocated hardware resources), and the name of the managed system in which the partition was defined.

Flags

Item	Description
-A	Returns all adapters of the particular type that are specified with the **-t** flag.
-a	Specifies the network IP addresses when the server, client, and gateway are IPv6 addresses.
-B *Image_filename*	Specifies the file name of the network boot image. The **-B** flag is a required flag for IPv6 addresses.
-C *Client*	Specifies the IP address of the partition to start the network.
-D	Performs a ping test to identify and use the adapter that can successfully ping the server that is specified with the **-S** flag.
-d *Duplex*	Specifies the duplex setting of the partition that is specified with the **-C** flag. The valid values for the **-d** flag are *full*, *half*, and *auto*.
-E	Specifies the setting for the environment variable. The following commands return the same output: **-E** *LPAR_NETBOOT_DEBUG=1* **export** *LPAR_NETBOOT_DEBUG=1*
-f	Force closes a virtual terminal session for the partition.
-G *Gateway*	Specifies the gateway IP address of the partition that is specified with the **-C** flag.
-g *args*	Specifies generic arguments for starting the partition. You can specify additional arguments with the firmware boot command, by using the **-g** flag. The **-g** flag is added for starting the preboot execution environment (PXE). An example for the **-g** argument follows: **-g** *autoyast=* *nfs://9.184.115.219//* *csminstall/csm/SLES10/09B873DC* *dhcptimeout=150* *install=nfs://9.184.115.219//* *csminstall/Linux/SLES/10/ppc64/GA/CD1*
-i	Forces an immediate shutdown of the partition. If this option is not specified, a delayed shutdown is performed.
-K *subnetmask*	Specifies the mask that the gateway uses in determining the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes, as in the IP 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.
-l *phys_loc*	Specifies the physical location code of the network adapter to use for network boot.
-M	Displays the network adapter MAC address and physical location code.
-m *maddress*	Specifies the MAC address of the network adapter to use for network boot.
-n	Instructs the partition to not network boot.
-S *Server*	Specifies the IP address of the partition, from which to retrieve the network boot image during network boot.
-s *Speed*	Specifies the speed setting of the partition that is specified with the **-C** flag.
-T	Enables or disables the display of the firmware-spanning tree. The valid values for the **-d** flag are *on* and *off*.
-t *ent*	Specifies the type of adapter for displaying the MAC address or physical location code discovery, or for network boot. The only valid value for the **-t** flag is *ent* for Ethernet.
-V *vlan_tag*	Specifies the VLAN tag identifier for tagging Ethernet frames during network installation for virtual network communication. The valid values for the **-V** flag are *0 - 4094*.
-v	Displays additional information while the command is running.
-x	Displays debug output while the command is running.
-Y *vlan_priority*	Specifies the VLAN tag priority for tagging Ethernet frames during network installation for virtual network communication. The valid values for the **-Y** flag are *0 - 7*.

Parameters

Item	Description
partition_name	Specifies the name of the partition.
partition_profile	Specifies the name of the partition profile to use.
managed_system	Specifies the name of the managed system on which the partition is defined.

Environment variables

Item	Description
INSTALLIOS_DEBUG	Prints the lpar_netboot debug output, when specified with the **installios** command.
LPAR_NETBOOT_DEBUG	Prints the lpar_netboot debug output. Hence, it is similar to the **-x** flag.
LPAR_NETBOOT_DEBUG_BOOT	Initiates the firmware boot command, when specified with the **-s** flag.
LPAR_NETBOOT_ADD_TIMEOUT	Extends the timeout value by 5 seconds, as shown in the following example: `LPAR_NETBOOT_ADD_TIMEOUT=5`
LPAR_NETBOOT_SUB_TIMEOUT	Lowers the timeout value by 8 seconds, as shown in the following example: `LPAR_NETBOOT_SUB_TIMEOUT=8`
LPAR_NETBOOT_SPANNING_TREE	Enables or disables the display of the firmware-spanning tree. The valid values for the **-d** flag are *on* and *off*. Hence, it is similar to the **-T** flag
OPEN_DEV_DEBUG	Displays the firmware open_dev debug output, when the value of the *OPEN_DEV_DEBUG* variable is set to *yes*.
FIRMWARE_DUMP	Displays the firmware dump for firmware debugging, when the value of the *FIRMWARE_DUMP* variable is set to *yes*.

Exit Status

Item	Description
0	Successful completion

Security

Access Control: You must have root authority to run the **lpar_netboot** command.

Examples

1. To retrieve MAC address and physical location code for partition `machA` with a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -M -n -t ent "machA" "machA_prof" "test_sys"
   ```

2. To network boot partition `machA` with a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -s auto -d auto -S 9.3.6.49 -G 9.3.6.1 -C 9.3.6.234
                "machA" "machA_prof" "test_sys"
   ```

3. To network boot partition `machA` with a specific MAC address of `00:09:6b:dd:02:e8` and a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -m 00096bdd02e8 -s auto -d auto -S 9.3.6.49 -G 9.3.6.1
                -C 9.3.6.234 "machA" "machA_prof" "test_sys"
   ```

4. To network boot partition `machA` with a specific physical location code of `U1234.121.A123456-P1-T6` and a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

   ```
   lpar_netboot -t ent -l U1234.121.A123456-P1-T6 -s auto -d auto -S 9.3.6.49
                -G 9.3.6.1 -C 9.3.6.234 "machA" "machA_prof" "test_sys"
   ```

5. To perform a ping test and a network boot of partition `machA` with a partition profile `machA_prof` on a managed system `test_sys`, enter the following command:

```
lpar_netboot -t ent -D -s auto -d auto -S 9.3.6.49 -G 9.3.6.1 -C 9.3.6.234
             "machA" "machA_prof" "test_sys"
```

6. To perform a ping test and a network boot of partition `machA` with a partition profile `machA_prof` on a managed system `test_sys` and disable firmware-spanning tree discovery, enter the following command:

```
lpar_netboot -t ent -T off -D -s auto -d auto -S 9.3.6.49 -G 9.3.6.1
             -C 9.3.6.234 "machA" "machA_prof" "test_sys"
```

Location

/opt/ibm/sysmgt/dsm/dsmbin/lpar_netboot

---

lparstat Command

Purpose

Reports logical partition (LPAR) related information and statistics.

Syntax

lparstat {**-i** [**-W** | **-s** | **-P**] | **-W** | **-s** | **-P** | **-d** | **-m** [**-e** [**r** | **R**] [**-p**[**w**]]] | [**-H** | **-h**] | [**-X** [**-o** *filename*]] [**-c**] | [**-E** [**w**]] [**-t**] [*Interval* [*Count*]] }

Description

The **lparstat** command provides a report of LPAR related information and utilization statistics. This command provides a display of current LPAR related parameters and Hypervisor information, as well as utilization statistics for the LPAR. An interval mechanism retrieves numbers of reports at a certain interval.

The various options of **lparstat** command are exclusive of each other. The **lparstat** command with no options will generate a single report containing utilization statistics related to the LPAR since boot time. If the **-h** option is specified, the report will include summary statistics related to the Hypervisor. If an *interval* and *count* are specified, the above report display repeats for every *interval* seconds and for *count* iterations. *interval* and *count* cannot be used with the **-i** option. Only root users can run the **-h** and **-H** flags.

The *interval* parameter specifies the amount of time in seconds between each report. If you do not specify the *interval* parameter, the **lparstat** command generates a single report that contains statistics for the time since system startup and then exits. You can specify the *count* parameter only with the *interval* parameter. If you specify the *count* parameter, its value determines the number of reports that are generated and the number of seconds apart. If you specify the *interval* parameter without the *count* parameter, reports are continuously generated. Do not specify a value of zero to the *count* parameter.

When the **lparstat** command is invoked without the **-i** flag, two rows of statistics are displayed. The first row displays the System Configuration, which is displayed once when the command starts and again whenever there is a change in the system configuration. The second row contains the Utilization Statistics which will be displayed in intervals and again any time the values of these statistics are deltas from the previous interval.

If you specify the **-X** option, the **lparstat** command creates an XML file.

The following information is displayed in the system configuration row:

type Indicates the partition type. The value can be either dedicated or shared.

mode Indicates whether the partition processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating.

smt Indicates whether simultaneous multithreading is enabled or disabled in the partition. If there are two SMT threads, the row is displayed as "on." However, if there are more than two SMT threads, the number of SMT threads is displayed.

lcpu Indicates the number of online logical processors.

mem Indicates online memory capacity.

Note: If Active Memory™ Expansion is enabled, **mem** specifies the expanded memory size configured for this LPAR. However, if the environment variable *AME_MEMVIEW* is set to **TRUE**, the **mem** value specifies the true memory size.

psize Indicates the number of online physical processors in the pool.

ent Indicates the entitled processing capacity in processor units. This information is displayed only if the partition type is shared.

If you specify the **-m** flag, the following information is displayed in the system configuration row:

lcpu Indicates the number of online logical processors.

ent Indicates the entitled processing capacity in processor units.

mem Indicates online memory capacity.

Note: If Active Memory Expansion is enabled, **mem** specifies the expanded memory size configured for this LPAR. However, if the environment variable *AME_MEMVIEW* is set to **TRUE**, the **mem** value specifies the true memory size.

mpsz Indicates the memory pool size of the pool that the partition belongs to (in GB).

iome Indicates the I/O memory entitlement of the partition (in MB).

iomp Indicates the number of I/O memory entitlement pools in the LPAR.

If you specify the -c flag, the following additional information is displayed in the system configuration row:

mmode
Indicates the system's memory mode. The values for **mmode** are:

Item	Description
Ded	Neither Active Memory Sharing nor Active Memory Expansion is enabled
Shar	Active Memory Sharing is enabled
Ded-E	Active Memory Expansion is enabled
Shar-E	Both Active Memory Sharing and Active Memory Expansion are enabled

mem Indicates the expanded memory size of the LPAR.

tmem Indicates the true memory size of the LPAR.

The following information is displayed in the utilization row:

%user Indicates the percentage of the entitled processing capacity used while executing at the user level (application).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%sys Indicates the percentage of the entitled processing capacity used while executing at the system level (kernel).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%idle Indicates the percentage of the entitled processing capacity unused while the partition was idle and did not have any outstanding disk I/O request.

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

%wait

Indicates the percentage of the entitled processing capacity unused while the partition was idle and had outstanding disk I/O request(s).

For dedicated partitions, the entitled processing capacity is the number of physical processors.

For uncapped partitions with a current physical processor consumption above their entitled capacity, the percentage becomes relative to the number of physical processor consumed (physc).

The following statistics are displayed when the partition type is shared or dedicated-donating:

physc Indicates the number of physical processors consumed.

vcsw Indicates the number of virtual context switches that are virtual-processor hardware preemptions.

The following statistics are displayed only when the partition type is shared:

%entc Indicates the percentage of the entitled capacity consumed. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

lbusy Indicates the percentage of logical processor(s) utilization that occurred while executing at the user and system level.

app Indicates the available physical processors in the shared pool.

phint Indicates the number of phantom (targeted to another shared partition in this pool) interruptions received.

The following statistics are displayed only when the **-h** flag is specified:

%hypv

Indicates the percentage of physical processor consumption spent making hypervisor calls.

hcalls Indicates the average number of hypervisor calls that were started.

The following statistic is displayed only if the hardware can use the SPURR, and the processor is not running at nominal speed:

%nsp Indicates the current average processor speed as a percentage of nominal speed.

The following statistic is displayed only if the turbo-mode accounting is disabled:

%utcyc

Indicates the total percentage of unaccounted turbo cycles.

The following statistics are displayed only when the **-d** flag is specified.

%utuser
Indicates the percentage of unaccounted turbo cycles in the user mode execution (application).

%utsys
Indicates the percentage of unaccounted turbo cycles in the kernel mode execution (kernel).

%utidle
Indicates the percentage of the unaccounted turbo cycles when the partition is idle and does not have any outstanding disk I/O requests.

%utwait
Indicates the percentage of the unaccounted turbo cycles when the partition is idle and has outstanding disk I/O requests.

If you specify the **-m** flag, the following metrics are displayed:

physb Indicates that the physical processor is busy.

%entc Indicates the percentage of the entitled capacity consumed. Because the time base over which this data is computed might vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals.

vcsw Indicates the number of virtual context switches that are virtual-processor hardware preemptions.

hpi Indicates the number of hypervisor page-ins occurred.

hpit Indicates the time that is spent waiting for hypervisor page-ins in milliseconds.

pmem Indicates the physical memory that is allocated to the LPAR by hypervisor in GB.

iomin Indicates the minimum entitlement of the I/O memory pool in MB.

iomu Indicates the I/O memory entitlement of the LPAR in use in MB.

iomf Indicates the free I/O memory entitlement in MB.

iohwm
Indicates the high water mark of I/O memory entitlement usage in GB.

iomaf Indicates the total number of times that allocation requests for I/O memory entitlement pools have failed since system startup.

If you specify the **-e** flag with the **-m** flag, the following information about I/O memory entitlement pools is displayed:

iompn Indicates the name of the I/O memory entitlement pool in MB.

iomin Indicates the minimum entitlement of the I/O memory pool in MB.

iodes Indicates the desired entitlement of the I/O memory pool in MB.

ioinu Indicates the entitlement of the I/O memory pool in use in MB.

iores Indicates the reserved entitlement of the I/O memory pool in MB.

iohwm
Indicates the high water mark of entitlement usage of the I/O memory pool in MB.

ioafl Indicates the total number of times that allocation requests for this I/O memory entitlement pool have failed since system startup.

The following statistics are displayed only when the -c flag is specified:

%xcpu
Indicates the percentage of utilization (relative to the overall CPU consumption by the logical partition) for the Active Memory Expansion (AME) activity.

xphysc
Indicates the number of physical processors used for the Active Memory Expansion activity.

dxm Indicates the size of the expanded memory deficit for the LPAR in MB.

pgcol Indicates the logical real memory pages of the calling partition in megabytes that are coalesced during the active memory sharing activity.

mpgcol
Indicates the number of megabytes of the memory pages that are called by the memory pool of the coalesced partition during the Active Memory sharing activity. If the partition is not authorized to access the poolwide statistics, the metric shows zero.

ccol Indicates the fraction of the CPU consumed in coalescing pages during the Active Memory sharing activity. If the partition is not authorized to access the poolwide statistics, the metric shows zero.

Note: Memory page coalescing is a transparent operation wherein the hypervisor detects duplicate pages, directs all the user read pages to a single copy, and reclaims the other duplicate physical memory pages.

Flags

Item	Description
-c	Adds the memory compression statistics of the LPAR to the default **lparstat** output. **Note:** This option is available only when Active Memory Expansion is enabled.
-d	Shows the detailed CPU utilization statistics. When the turbo-mode accounting is disabled, the **lparstat** command shows the breakdown by category of the unaccounted turbo cycles along with the dedicated, donating or shared utilization columns: %user, %sys, %idle, %wait, %entc, %idon, %bdon, %istol and %bstol.
-e	Displays information about the I/O memory entitlement pools of the LPAR. You can specify the **-e** flag only with the **–m** flag. See the metrics that are displayed when you specify the **-m** flag.
-E	Reports Scaled Processor Utilization Resource Register (SPURR) based utilization metrics if run on a SPURR-capable processor.
-h	Adds summary hypervisor statistics to the default **lparstat** output.
-H	Provides detailed Hypervisor information. This option basically displays the statistics for each of the Hypervisor calls. The various Hypervisor statistics displayed by this option, for each of the Hypervisor calls, are as below: **Statistic Description** **Number of calls** Number of Hypervisor calls made. **Total Time Spent** Percentage of total time spent in this type of call. **Hypervisor Time Spent** Percentage of Hypervisor time spent in this type of call. **Average Call Time** Average call time for this type of call in nano-seconds. **Maximum Call Time** Maximum call time for this type of call in nano-seconds.

Item	Description
-i	Lists details on the LPAR configuration. The various details displayed by the **-i** option are listed below:

Name **Description**

Partition Name
Logical partition name as assigned at the HMC.

Partition Number
Number of this logical partition.

Power Save Mode
Power saving mode of this logical partition.

Online Virtual CPUs
Number of CPUs (virtual engines) currently online.

Maximum Virtual CPUs
Maximum possible number of CPUs (virtual engines).

Online Memory
Amount of memory currently online.

Maximum Memory
Maximum possible amount of Memory.

Type Indicates whether the LPAR is using dedicated or shared CPU resource and if the SMT is turned ON. The Type is displayed in the format [Shared | Dedicated] [-SMT] [-#]

The following list explains the different Type formats:

- Shared - Indicates that the LPAR is running in the Shared processor mode.
- Dedicated - Indicates that the LPAR is running in the dedicated processor mode.
- SMT[-#] - Indicates that the LPAR has SMT mode turned ON and the number of SMT threads is 2. If the number of threads is greater than 2, then the number of threads is also displayed.

Mode Indicates whether the LPAR processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating.

Entitled Capacity
The number of processing units this LPAR is entitled to receive.

Variable Capacity Weight
The priority weight assigned to this LPAR which controls how extra (idle) capacity is allocated to it. A weight of -1 indicates a soft cap is in place.

Minimum Capacity
The minimum number of processing units this LPAR was defined to ever have. Entitled capacity can be reduced down to this value.

Maximum Capacity
The maximum number of processing units this LPAR was defined to ever have. Entitled capacity can be increased up to this value.

Capacity Increment
The granule at which changes to Entitled Capacity can be made. A value in whole multiples indicates a Dedicated LPAR.

Maximum Physical CPUs in System
The maximum possible number of physical CPUs in the system containing this LPAR.

Item	Description
	(*Details displayed by the* **-i** *flag, are as follows*):
	Active Physical CPUs in System The current number of active physical CPUs in the system containing this LPAR.
	Active CPUs in Pool The maximum number of CPUs available to this LPAR's shared processor pool.
	Shared Physical CPUs in system The number of physical CPUs available for use by shared processor LPARs.
	Maximum Capacity of Pool The maximum number of processing units available to this LPAR's shared processor pool.
	Entitled Capacity of Pool The number of processing units that this LPAR's shared processor pool is entitled to receive.
	Unallocated Capacity The sum of the number of processor units unallocated from shared LPARs in an LPAR group. This sum does not include the processor units unallocated from a dedicated LPAR, which can also belong to the group. The unallocated processor units can be allocated to any dedicated LPAR (if it is greater than or equal to 1.0) or shared LPAR of the group.
	Physical CPU Percentage Fractional representation relative to whole physical CPUs that these LPARs virtual CPUs equate to. This is a function of Entitled Capacity / Online CPUs. Dedicated LPARs would have 100% Physical CPU Percentage. A 4-way virtual with Entitled Capacity of 2 processor units would have a 50% physical CPU Percentage.
	Minimum Memory Minimum memory this LPAR was defined to ever have.
	Minimum Virtual CPUs Minimum number of virtual CPUs this LPAR was defined to ever have.
	Unallocated Weight Number of variable processor capacity weight units currently unallocated within the LPAR group.
	Partition Group ID LPAR group that this LPAR is a member of.
	Shared Pool ID Identifier of Shared Pool of Physical processors that this LPAR is a member.

Item	Description
	(*Details displayed by the* **-i** *flag, are as follows*): **Memory Mode** Indicates whether the memory mode is shared or dedicated. If Active Memory Expansion is enabled, the memory mode also includes a new mode called **Expanded**. **Total I/O memory entitlement** The I/O memory entitlement of the LPAR. **Variable memory capacity weight** The variable memory capacity weight of the LPAR. **Memory Pool ID** The memory pool ID of the pool that the LPAR belongs to. **Physical Memory in the Pool** The physical memory present in the pool that the LPAR belongs to. **Hypervisor Page Size** The page size that hypervisor uses for the page-in and page-out of LPAR logical-memory pages. **Unallocated Variable Memory Capacity Weight** The unallocated variable memory-capacity weight of the LPAR. **Unallocated I/O memory entitlement** The unallocated I/O memory entitlement of the LPAR. **Memory Group ID of LPAR** The memory group ID of the Workload Manager group that the LPAR belongs to. **Target Memory Expansion Factor** The target memory expansion factor configured for the LPAR. **Note:** The target memory expansion factor is displayed when Active Memory Expansion is enabled. **Target Memory Expansion Size** The target expanded memory size for the LPAR. The target expanded memory size is the true memory size multiplied by the target memory expansion factor. **Note:** The target memory expansion size is displayed when Active Memory Expansion is enabled. **Power Save Mode** The power saving mode for the LPAR. **Subprocessor Mode** The subprocessor mode for the LPAR. You can specify the -i flag alone or with the **-P**, **-W**, and **-s** flags.
-m	Displays the statistics that are related to the following aspects: • The logical memory • The physical memory backing the logical memory of the LPAR • The I/O memory entitlement of the LPAR • The memory pool information on the pool that the LPAR belongs to For more information about the metrics that are displayed when you specify the **-m** flag, see the metrics section.
-o	Specifies the file name for the XML output.
-p	Displays the information about the page coalescing statistics of the LPAR. You can specify the **-p** flag only with the **-m** flag. When you run the **lparstat** command with the **-w** and **-p** flags, the result displays all the metrics that are displayed by the **-e** flag in a single line.
-P	Displays information about the energy management tuning parameters. You can specify the **-P** flag alone or with the **-i**, **-W**, and **-s** flags.
-r	Resets the high water mark of I/O memory entitlement once at the beginning of the command. You can use this flag only with the **-m** and **-e** flags.
-R	Resets the high water mark at the beginning of each monitoring interval. If you specify both the **-r** and **-R** flags, the **-R** flag takes effect.

Item	Description
-s	Displays LPAR information. The **-s** flag displays the following details: **Service partition ID** The service partition ID as assigned by the Hardware Management Console (HMC). **Number of configured LPARs** The number of LPARs that are configured on the HMC. You can specify the **-s** flag alone or with the **-P**, **-i**, and **-W** flags.
-t	Displays the time in the *HH:MM:SS* format when the command is run with intervals.
-W	Lists details of the workload partition (WPAR) configuration. If the command is run from the global environment, the **WPAR Key** value is 0. The **-W** flag displays the following details: **WPAR Key** WPAR static identifier. **WPAR Configured ID** WPAR dynamic identifier. **WPAR Maximum CPUs** Number of processors in a resource set. It displays the value of 0 if it is not restricted. **WPAR Effective CPUs** Number of processors in an effective resource set. It displays the value of 0 if it is not restricted. **WPAR CPU Percentage** WPAR processor-limit percentage. You can specify the **-W** flag alone or with **-P**, **-i**, and **-s** flags.
-X	Generates the XML output. The default file name is **lparstat_DDMMYYHHMM.xml**, unless the user specifies a different file name with the **–o** option.

Note: If Pool Utilization Authority (PUA) is not available, the **app** column is not displayed.

Examples

1. To get the default LPAR statistics, enter the following command:
   ```
   lparstat 1 1
   ```
2. To get default LPAR statistics with summary statistics on Hypervisor, enter the following command:
   ```
   lparstat -h 1 1
   ```
3. To get the information about the partition, enter the following command:
   ```
   lparstat -i
   ```
4. To get detailed Hypervisor statistics, enter the following command:
   ```
   lparstat -H 1 1
   ```
5. To get statistics about the shared memory pool and the I/O memory entitlement of the partition, enter the following command:
   ```
   lparstat -m
   ```
6. To get statistics about I/O memory pools inside the LPAR, enter the following command:
   ```
   lparstat -me
   ```
7. If the LPAR is running in shared mode and with 4 SMT threads the type would be in the following format:
   ```
   Type - Shared-SMT-4
   ```
8. If the LPAR is running in dedicated mode and with 2 SMT threads the type would be in the following format:
   ```
   Type - Dedicated-SMT
   ```
9. To calculate the memory compression statistics in an LPAR when Active Memory Expansion is enabled, enter the following command:
   ```
   lparstat -c 1 1
   ```

10. To get statistics about page coalescing inside an LPAR, enter the following command:
    ```
    lparstat -mp
    ```

Files

Item	Description
/usr/bin/lparstat	Contains the **lparstat** command.

Related reference:
"mpstat Command" on page 901
"iostat Command" on page 111
Related information:
vmstat command
sar 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.

If you enter `lpc -?`, the system displays the command usage message and returns 0.

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**).

Item	Description
? [*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**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/spool/lp/*	
/var/spool/lp/system/pstatus	

Error Codes

Item	Description
?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 reference:

"lpq Command" on page 319
"lpr Command" on page 323
"lprm Command" on page 327
"lpsched Command" on page 330

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:

Item	Description
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

Item	Description
-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

Item	Description
/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:

Remote Printing Overview

---

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.

If you enter `lpfilter -?`, the system displays the command usage message and returns 0.

Flags

Item	Description
- (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*

Item	Description
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.

Item	Description
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 the **ed** command.

lp flag	Properties
-T	**Characteristic** Content type (input) **Keyword** INPUT **Possible patterns** content-type
N/A	**Characteristic** Content type (output) **Keyword** OUTPUT **Possible patterns** content-type
N/A	**Characteristic** Printer type **Keyword** TERM **Possible patterns** printer-type
-d	**Characteristic** Printer name **Keyword** PRINTER **Possible patterns** printer-name
-f, -o cpi=	**Characteristic** Character pitch **Keyword** CPI **Possible patterns** integer

lp flag	Properties
-f, -o lpi=	**Characteristic** Line pitch **Keyword** LPI **Possible patterns** integer
-f, -o length=	**Characteristic** Page length **Keyword** LENGTH **Possible patterns** integer
-f, -o width=	**Characteristic** Page width **Keyword** WIDTH **Possible patterns** integer
-P	**Characteristic** Pages to print **Keyword** PAGES **Possible patterns** page-list
-S	**Characteristic** Character set Print wheel **Keyword** CHARSET CHARSET **Possible patterns** character-set-name print-wheel-name
-f	**Characteristic** Form name **Keyword** FORM **Possible patterns** form-name
-y	**Characteristic** Modes **Keyword** MODES **Possible patterns** mode

lp flag	Properties
-n	**Characteristic** Number of copies **Keyword** COPIES **Possible patterns** 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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"lpadmin Command" on page 277

Related information:

ed 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. 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.

If you enter `lpforms -?`, the system displays the command usage message and returns 0.

With the first **lpforms** command line, one of the following flags must be used:

Flags

Item	Description
- (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:

Item	Description
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.

Item	Description
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:

Item	Description
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**.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"lpadmin Command" on page 277

"lp Command" on page 266

"lpstat Command" on page 331

Related information:

terminfo command

lphistory Command

Purpose

Displays or clears the history list of least-privilege (LP) commands that have been run during the current resource monitoring and control (RMC) session.

Syntax

- To list a particular number of previously-issued commands:
  - On the local node:

    lphistory [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]
  - On all nodes in a domain:

    lphistory -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]

- On a subset of nodes in a domain:

  lphistory -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]

- To clear the history list:
  - On the local node:

    lphistory -c [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

  - On all nodes in a domain:

    lphistory -c -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

  - On a subset of nodes in a domain:

    lphistory -c -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

Description

The **lphistory** command lists the history of LP commands that have been run by the least-privilege resource manager. The command history is maintained as records in the RSCT audit log. By default, only the command string (the path name plus arguments) from each audit log record is listed. The **-L** flag controls the output format of **lphistory**; use it to display specific fields as needed. The selection flags (**-B**, **-C**, **-E**, **-m**, **-S**, or **-u**) control the selection string that is passed to **lsaudrec**.

The **lphistory** command takes one optional parameter: the number of records to list. The default value of *num_records* is 10. If none of the selection flags is used, the latest number of records in the audit log (specified by *num_records*) are listed. Otherwise, the latest number of records (specified by *num_records*) from those selected by one or more of the selection flags are listed. This selection process applies to the audit records on each node specified by the **-a** flag or the **-n** flag. If neither **-a** nor **-n** is specified, the selection process applies to the audit records on the local node.

The **-B** and **-E** flags take time stamps as arguments. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day of the month (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year.

You can use the wild card character (%) with identity-related arguments (*user_ID*, *mapped_ID*) and command names. The % can be placed at the beginning or end of the string, or anywhere within it. You cannot use any wild card characters when specifying *command_path*.

You can remove audit log records using the **-c** flag. If none of the selection flags is specified, all audit log records for the least-privilege resource manager are removed. Otherwise, the records selected by one or more of the selection flags are removed. The **-c** flag cannot be used with the **-L** flag or the *num_records* parameter.

Flags

-a Displays previously-issued LP commands for all nodes in the domain.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

You cannot specify this flag with the **-n** flag.

-B *MMddhhmmyyyy*
Specifies a beginning time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or after this time.

–c Clears the history of LP commands. You cannot specify this flag with the *number_of_commands* parameter or the **-n** flag.

-C *command_name*
Specifies a command name. **lphistory -C** lists or removes only those records that contain *command_name*, which is the name of a command without a fully-qualified path (**mkrsrc**, for example). You can use wild card characters in *command_name*.

-E *MMddhhmmyyyy*
Specifies an ending time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), mm is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or before this time.

-L a | c | e | m | n | t | u | x
By default, only the command string (path name plus arguments) from each audit log record is listed. If this flag is specified, the argument is one or more of the following letters; the fields are displayed in the same order as the letters in the flag argument.

a Displays all fields from the audit log in the following order: **t**, **u**, **m**, **n**, **x**, **c** (specifying **-L** a is the same as specifying **-L tumnxc**)

c Displays the command string (the default)

e Displays the standard error output

m Displays the mapped identity

n Displays the name of the node where the command ran

t Displays the time field

u Displays the authenticated user identity

x Displays the LP command exit status

You cannot specify this flag with the **-c** flag.

-m *mapped_ID*
Specifies a mapped identity. **lphistory -m** lists or removes only those records that contain *mapped_ID*. You can use wild card characters in *mapped_ID*.

-n *host1***[,***host2***,...]**
Specifies one or more nodes in the cluster on which the LP command history list is to be retrieved or cleared. (By default, the history list for the local node is retrieved or cleared.)

This flag is valid only in a management domain or a peer domain. If the **CT_MANAGEMENT_SCOPE** environment variable is not set, management domain scope is chosen first (if a management domain exists) and then peer domain scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds.

You cannot specify this flag with the **-a** flag.

-S *command_path*
Specifies a command path name. **lphistory -S** lists or removes only those records that contain *command_path*, which is identical to the value of the **CommandPath** in the LPCommands class (**/opt/rsct/bin/mkrsrc**, for example). You cannot use wild card characters in *command_path*.

-u *user_ID*
Specifies an authenticated user identity. **lphistory -u** lists or removes only those records that contain *user_ID*. You can use wild card characters in *user_ID*.

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

-T Writes the command's trace messages to standard error.

-V Writes the command's verbose messages to standard output.

Parameters

num_records
Specifies the number of commands to be displayed from the history list. You can list a minimum of one command and a maximum of 100 commands. The default value is 10. You cannot specify this parameter with the **-c** flag.

Security

To run the **lphistory** command, you need write permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. 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 LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-V** flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list 20 LP commands that were previously issued on the local node, enter:
   ```
   lphistory 20
   ```
2. Suppose **nodeA** is in a management domain and **CT_MANAGEMENT_SCOPE** is set to **3**. To list the LP command history on **nodeA**, enter:
   ```
   lphistory -c -n nodeA
   ```
3. To display the last 15 LP commands invoked with time, user ID, mapped ID, mechanism, return code, standard error, command name, and command string, enter:
   ```
   lphistory -L a 15
   ```
4. To display the LP command names that end with `rsrc`, enter:
   ```
   lphistory -C %rsrc
   ```
5. To display the LP commands that were invoked after 11:30 PM on April 18, 2006, enter:
   ```
   lphistory -B 041823302006
   ```

Location

/opt/rsct/bin/lphistory
Contains the **lphistory** command.

---

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 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**.

If you enter `lpmove -?`, the system displays the command usage message and returns 0.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/spool/lp/*	

Related reference:

"lp Command" on page 266

"lpadmin Command" on page 277

"lpstat Command" on page 331

Related information:

accept command

---

lppchk Command

Purpose

Verifies files of an installable software product.

Syntax

lppchk [**-R** { **Path** | **ALL** }] { **-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

Item	Description
-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.
-R { **Path | ALL** }	Indicates a user-specified installation location.
-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

Item	Description
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.

Exit Status

Item	Description
0 (zero)	The command completed successfully.
nonzero	An error was found.

The **lppchk** command returns zero if no errors were found. Any other return value indicates an error was found.

Note: If **lppchk -f** (size) or **lppchk -c** (checksum) detects a mismatch in the respective size or checksum for a file, it does not report an error for the file if the file has been changed by an interim fix within **/usr/emgrdata/DBS/files.db**.

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

Item	Description
/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 reference:

"installp Command" on page 62

Related information:

sum command

tcbck command

lppmgr Command

Purpose

Manages an existing installp image source.

Syntax

lppmgr -d *DirectoryOrDevice* **[-r** | **-m** *MoveDirectory* | **-a** *APAR number* **]** { **[-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 that are the same level as bases of the same file set. Such updates can create conflicts that lead to installation failure (**-u** Flag).
4. Remove message and locale file sets other than the language you specify (**-k** Flag).
5. Remove superseded file sets (**-x** Flag).
6. Remove non-system images from a NIM lpp_source resource (**-X** Flag).

By default, **lppmgr** lists all images that are filtered by the preceeding 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 file sets. It is also not intended to address any issues other than those mentioned earlier. Before you use the "**-X**" flag, you must have a good understanding of NIM, system images (known as SIMAGES in NIM), and the workings of a NIM **lpp_source** resource.

Flags

Item	Description
-a *APAR number*	Displays file sets associated with an APAR number or keyword. If there are more than one APAR numbers, they must be contained in quotation marks and separated by spaces.
-b	Causes **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 it can be any directory, NFS mount point, or cdrom device. If the directory is not writable, you must use the **-t** flag. If the target of your operation is a NIM **lpp_source** resource, you must 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 that is 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** removes all filtered files without further user interaction. This flag cannot be used with the "**-m**" flag.
-s	Prints space usage information. This flag prints the amount of space a particular file set is using and the total amount of space in question. Some buffer space is added for file metadata.
-t	Specifies that **lppmgr** does *not* rebuild the **.toc** file. This flag can be useful for having a quick look without having to rebuild the entire **.toc** file, which can take some time. Also, this flag is required for read-only devices.
-u	Causes **lppmgr** to filter for duplicate updates and conflicting updates that are the same level as bases of the same file set.
-V	Specifies verbose mode. **lppmgr** gives 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 that are 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 the following command:

   ```
   lppmgr -d /myimages -u
   ```

2. To remove all duplicate and conflicting updates in image source directory **/myimages**, enter the following command:

   ```
   lppmgr -d /myimages -u -r
   ```

3. To remove all duplicate and conflicting updates, duplicate base levels, and all message/locale file sets other than "en_US" in prompted mode, enter the following command:
   ```
   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 the following command:
   ```
   lppmgr -d /lpps/433 -x -X -m  /backups
   ```
5. To list all the file sets associated with APAR numbers IX38794 and IX48523 in image source directory **/myimages**, enter the following command:
   ```
   lppmgr -d /myimages -a "IX38794 IX48523"
   ```

Files

Item	Description
/usr/lib/instl/lppmgr	Contains the **lppmgr** command.

Related reference:

"installp Command" on page 62

"lslpp Command" on page 432

"lsnim Command" on page 469

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

Item	Description
-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. 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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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, enter:

   ```
   lpq  -W
   ```

Files

Item	Description
/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.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"lpc Command" on page 297

"lpr Command" on page 323

"lprm Command" on page 327

Related information:

qchk command

qconfig command

Print spooler

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

Item	Description
-# *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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/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.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 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 reference:
"lpd Command" on page 299
"lpq Command" on page 319
"lpc Command" on page 297
"lprm Command"

Related information:
pr command
qdaemon command
Printing administration
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

Item	Description
-	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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/lprm	Contains the **lprm** command.
/etc/qconfig	Contains the configuration file.

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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:
"lpq Command" on page 319
"lpr Command" on page 323
"lp Command" on page 266
"lpsched Command"
Related information:
qcan 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.

If you enter `lpsched -?`, the system displays the command usage message and returns 0.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/var/spool/lp/*	

Related reference:

"lpadmin Command" on page 277

Related information:

accept command

enable 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

Item	Description
-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. **Note:** You cannot use both the **-p** flag and the **-t** flag at the same time.
-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 the status information with longer queue names, device names, and job numbers. This flag cannot be used with the **-t** flag. If the **-t** flag and the **-W** flag are used simultaneously, the first specified flag takes precedence. If the **-W** flag and **-l** flag are used simultaneously, the result displays the long status of the print job in the semicolon-separated format.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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, enter:

```
lpstat  -W
```

Files

Item	Description
/var/spool/lpd/*	Contains temporary copies of remote enqueued files.

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.

The command **lpstat -o** *printername* is used to list all the requests queued on the specified printer. If *printername* points to a remote printer, 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.

If you enter `lpstat -?`, the system displays the command usage message and returns 0.

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**.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related information:

disable command

enable command

qchk command

Print spooler

---

lpsystem Command

Purpose

Registers remote systems with the print service.

Note: This is a System V Print Subsystem command.

Syntax

lpsystem [*-t Type*] [**-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.

If you enter `lpsystem -?`, the system displays the command usage message and returns 0.

Flags

Item	Description
-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*	Allows you to associate a free-form comment with the system entry.
-t *Type*	Specifies the type of remote system. The only supported value for the -t flag is bsd.

Security

Only a user with appropriate privileges may execute the **lpsystem** command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Table 4. Files

Item	Description
/etc/lp/*	**/var/spool/lp/***

Related reference:
"lpsched Command" on page 330

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 reference:

"lp Command" on page 266

"lpstat Command" on page 331

Related information:

cancel command

enable 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, 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.

If you enter `lpusers -?`, the system displays the command usage message and returns 0.

Parameters

Item	Description
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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"lp Command" on page 266

ls Command

Purpose

Displays the contents of a directory.

Syntax

To Display Contents of Directory or Name of File

ls [-1] [-A] [-C] [-E] [-F] [-H | -L] [-N] [-R] [-S] [-X] [-a] [-b] [-c] [-d] [-e] [-f] [-g] [-i] [-k] [-l] [-m] [-n] [-o] [-p] [-q] [-r] [-s] [-t] [-u] [-U] [-x] [*File ...* **]**

To Display Contents of Directory

ls -f [-C] [-d] [-i] [-m] [-s] [-X] [-x] [-1] [-U] [*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.

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 **-U** flag is the same as with the **-l** flag, except for the addition of an 11th character interpreted as follows:

Item	Description
E	Indicates a file has extended attributes (EA) information. The EA of a file is displayed by using the **getea** command.
-	Indicates a file does not have extended attributes information.
e	Indicates a file is encrypted.

Encryption takes precedence over the presence of the Access Control Lists (ACLs) and other EAs.

The mode displayed with the **-e** and **-l** flags is interpreted as follows:

If the first character is:

Item	Description
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:

Item	Description
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:

Item	Description
+	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

Item	Description
-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 has no effect if it is not 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 extent attributes of a file that exists in Vxfs file systems, such as space reservation, fixed extent size, and extent allocation flag information. You must specify the **-l** flag with this flag; otherwise, the **ls** command ignores the **-E** flag and completes the execution.
-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. Symbolic links that are named as operands are not followed unless you have specified the **-H** or **-L** flag.
-g	Displays the same information as the **-l** flag, except the **-g** flag suppresses display of the owner and symbolic link information.

Item	Description
-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. If the file system has an internal snapshot, the **.snapshot** directory and all its contents do not have unique i-node numbers.
-k	Sets the block size for **-s** option and the per-directory block count written for **-l**, **-n**, **-g**, and **-o** options to 1024 bytes.
-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.
-S	Sorts with the primary key being file size (in decreasing order) and the secondary key being file name in the collating sequence (in increasing order).
-t	Sorts by time of last modification (latest first) instead of by name. For a symbolic link, the time used as the sort key is that of the symbolic link itself.
-U	Displays similar information as the **-l** flag. Displays the mode (including security information, named extended attribute information and encryption 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.
-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 multi-column format.
-X	Prints long user names when used with other flags that display user names. The upper limit is determined by the **max_logname** ODM attribute in the PdAt and CuAt object classes. If a user name is greater than the **max_logname** attribute, it will be truncated to the number of characters as specified by the **max_logname** attribute, less one character.
-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:

Item	Description
0	All files were written successfully.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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.
5. To display detailed information with expanded user and group name, type:

   ```
   ls  -lX .profile
   ```

   This displays a long listing with detailed information about **.profile**.
6. To display data about whether extended attributes are set for the files in the current directory, type:

   ```
   ls -U
   ```

   - For releases AIX 5.3 and earlier:

     Example output:

     ```
     -rwSr-x---+   1 root     system           28 Apr 29 03:23 only_aixc
     -rwSr-x---E   1 root     system            4 Apr 29 03:23 only_aixc_ea
     -rw-r--r--E   1 root     system            4 Apr 29 03:23 only_ea
     ----------+   1 root     system          265 Apr 29 03:23 only_nfs4
     ----------E   1 root     system           64 Apr 29 03:23 only_nfs4_ea
     -rw-r--r---   1 root     system            4 Apr 29 03:23 only_regular
     ```

   - For releases AIX 6.1 and later:

     Example output:

     ```
     -rwSr-x---+    1 root     system           28 Apr 29 03:23 only_aixc
     -rwSr-x---E    1 root     system            4 Apr 29 03:23 only_aixc_ea
     -rw-r--r--E    1 root     system            4 Apr 29 03:23 only_ea
     ```

```
----------+    1 root     system          265 Apr 29 03:23 only_nfs4
----------E    1 root     system           64 Apr 29 03:23 only_nfs4_ea
-rw-r--r---    1 root     system            4 Apr 29 03:23 only_regular
-rwxrwxr-xe    2 root     system          256 May 25 16:27 encry_ex
```

Files

Item	Description
/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 reference:

"mkpasswd Command" on page 762

Related information:

find command

Files command

National Language Support Overview for Programming

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
- Authentication mechanism in use
- Attribute search mode
- Default user attribute entry location
- Timeout period (seconds) for LDAP client requests to the server

- User objectclass used in the LDAP server
- Group objectclass used in the LDAP server

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Example

1. 1.To list the status of the **secldapclntd** daemon, type:
   ```
   /usr/sbin/ls-secldapclntd
   ```

Files

Item	Description
/etc/security/ldap/ldap.cfg	Contains information needed by the **secldapclntd** daemon to connect to the server.

Related information:
mksecldap command
start-secldapclntd command
restart-secldapclntd command
/etc/security/ldap/ldap.cfg file

---

lsactdef Command

Purpose

Displays the action definitions of a resource or a resource class.

Syntax

To display the action definitions of a *resource:*

lsactdef [**-p** *property*] [**-s i** | **o**] [**-e**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*action1* [*action2 ...*]]

To display the action definitions of a *resource class:*

lsactdef -c [**-p** *property*] [**-s i** | **o**] [**-e**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*action1* [*action2 ...*]]

To display all resource class names:

lsactdef

Description

The **lsactdef** command displays a list of the action definitions of a resource or a resource class. By default, this command displays the action definitions of a *resource.* To see the action definitions of a *resource class,* specify the **-c** flag.

If you do not specify any actions on the command line, this command only displays actions that are defined as **public**. To override this default, use the **-p** flag or specify on the command line the names of the actions that have definitions you want 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, this command does not display action descriptions. To display action definitions and descriptions, specify the **-e** flag.

Flags

-c Displays the action definitions for *resource_class*.

-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 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 Specifies expanded format. Displays descriptions along with the action definitions.

-i Specifies input format. Generates a template of *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" format — 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 "OR"ed 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 table format. Each attribute is displayed in a separate column, with 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

resource_class
Specifies the name of the resource class with the action definitions that you want to display. If *resource_class* is not specified, a list of all of the resource class names is displayed.

action1 [*action2...*]
Specifies one or more actions. If *resource_class* is specified, zero or more action names can be specified. If no actions are specified, all of the action definitions for *resource_class* are displayed. Enter specific action names to control which actions are displayed and in what order. Use blank spaces to separate action names.

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 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect 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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

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_name      = "MatchCriteria"
 display_name      = ""
 description       = ""
        element_data_type = "char_ptr"
 element_index     = 0
sd_element 2:
        element_name      = "IncludeDetail"
 display_name      = ""
 description       = ""
        element_data_type = "uint32"
        element_index     = 1
```

Location

/opt/rsct/bin/lsactdef

---

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

Flag

Item	Description
-c	Causes colon format output for use by SMIT.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/lsallq	Contains the **lsallq** command.
/etc/qconfig	Configuration file.

Related reference:

"lsque Command" on page 487

"mkque Command" on page 780

Related information:

qconfig command

Listing print queues and print queue devices

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

Flags

Item	Description
-q *Name*	Specifies the queue name.
-c	Causes colon format output for use by SMIT.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/lsallqdev	Contains the **lsallqdev** command.
/etc/qconfig	Configuration file.

Related reference:

"lsquedev Command" on page 488

"lsallq Command" on page 349

Related information:

chquedev command

Listing print queues and print queue devices

---

lsarm command

Purpose

Displays Application Response Measurement (ARM) application and process usage information.

Syntax

lsarm -a [**-g**] [**-t**] [**-u**] [*ApplicationName* ...]

or

lsarm -p [**-a** [**-g**] [**-t**] [*ProcessID* ...]]

Description

The **lsarm** command displays information about applications registered with the operating system using the Application Response Measurement (ARM) APIs. The **-a** flag displays information about applications by application name. The **-p** option displays information about the applications used by a process.

Flags

Item	Description
-a	Displays application names.
-g	Displays group names associated with the application.
-p	Displays transaction names associated with the application.
-u	Displays process numbers using the application.
-t	Displays the applications used by a process.

Parameters

Item	Description
ApplicationName	Specifies a list of one or more applications for which the **lsarm** command should display information.
ProcessID	Specifies a list of one or more process IDs for which the **lsarm** command should display information.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To display all application names, type:
   ```
   lsarm -a
   ```
2. To display group names, transaction class names, and process numbers using the 'database' application, type:
   ```
   lsarm –a –g –t –u  database
   ```
3. To display the process IDs for all processes using ARM applications, type:
   ```
   lsarm –p
   ```
4. To display the application names, group names, and transaction names used by process 25038, type:
   ```
   lsarm –p –a –g –t 25038
   ```

Location

/usr/ewlm/sbin/lsarm

Related Information

---

lsassocmap Command

Purpose

Displays an association map.

Syntax

lsassocmap [**-c** *association_class*] [**-h**] [**-TV**] [*endpoint*...]

Description

The **lsassocmap** command displays the association classes available on a cluster, including the endpoints of each association. Names and endpoints of Common Information Model (CIM) association classes that are registered with the CIM resource manager are listed in table format, similar to the output of the **lscondresp** command.

If you specify the **lsassocmap** command without any parameters, it displays all of the association classes, endpoints, and roles. A *role* is the name of the class reference property in the association class definition. Roles can be used as parameters to the **-o** and **-R** flags of the **lsrsrcassoc** command to filter output. See "lsrsrcassoc Command" on page 513 for more information.

The **-c** flag limits the associations displayed to only those provided by a specific association class. You can specify any number of classes by using the *endpoint* parameter; only associations containing those classes as references (endpoints) are displayed.

Parameters

endpoint...
: Specifies one or more endpoint classes. Only association classes containing references to one of the *endpoint* classes are displayed.

Flags

-c *association_class*
: Displays associations for *association_class*.

-h
: Writes the command usage statement to standard output.

-T
: Writes the command trace messages to standard error. For your software service organization use only.

-V
: Writes the command verbose messages to standard output.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

When the **-T** flag is specified, this command trace messages are written to standard error.

Exit status

0
: The command ran successfully.

1
: An error occurred with RMC.

2
: An error occurred with the command-line interface (CLI) script.

3
: An incorrect flag was specified on the command line.

4
: An incorrect parameter was specified on the command line.

5
: An error occurred with RMC that was based on incorrect command-line input.

6
: The specified association class cannot be found.

Implementation specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack and the Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
/opt/rsct/bin/lsassocmap	

Examples

To display associations that are available in a cluster, enter the following command:

```
lsassocmap
```

The following output is displayed for the AIX platform:
The following output is displayed for other platforms:

```
Association Class             Role 1         Associator 1           Role 2        Associator 2          Node
cimv2.IBMAIX_RunningOS        Antecedent     IBMAIX_OperatingSystem Dependent     IBMAIX_ComputerSystem c175nf14
cimv2.IBMAIX_OSProcess        GroupComponent IBMAIX_OperatingSystem PartComponent IBMAIX_UnixProcess    c175nf14
cimv2.IBMAIX_CSProcessor      GroupComponent IBMAIX_ComputerSystem  PartComponent IBMAIX_Processor      c175nf14
cimv2.IBMAIX_HostedFileSystem GroupComponent IBMAIX_ComputerSystem  PartComponent CIM_FileSystem        c175nf14


Association Class             Role 1         Associator 1           Role 2        Associator 2          Node
cimv2.Linux_RunningOS         Antecedent     Linux_OperatingSystem  Dependent     Linux_ComputerSystem  c175nf14
cimv2.Linux_OSProcess         GroupComponent Linux_OperatingSystem  PartComponent Linux_UnixProcess     c175nf14
cimv2.Linux_CSProcessor       GroupComponent Linux_ComputerSystem   PartComponent Linux_Processor      c175nf14
cimv2.Linux_HostedFileSystem  GroupComponent Linux_ComputerSystem   PartComponent CIM_FileSystem       c175nf14
```

lsattr Command

Purpose

Displays attribute characteristics and possible values of attributes for devices in the system.

Syntax

lsattr { **-D** [**-O**] | **-E** [**-O**] | **-P** [**-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 specific device or type of device. If you do not specify the device logical name with the **-l** *Name* flag, you must use a combination of one or all of the **-c** *Class*, **-s** *Subclass*, and **-t** *Type* flags to uniquely identify the predefined device.

You must specify one of the following flags with the **lsattr** command:

Item	Description
-D	Displays default values.
-E	Displays effective values (valid only for customized devices that are specified with the **-l** flag).
-F *Format*	Specifies the user-defined format.
-P	Displays device values when the device was last configured.
-R	Displays the range of legal values.

When you display the effective values of the attributes for a customized device, the information is obtained from the Configuration database, not the device. The database values reflect how the device is configured unless it is reconfigured with the **chdev** command by using the **-P** or **-T** flag. If the reconfiguration occurs, the information that is 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 it is also used with the **-O** flag.

The **-P** flag displays the attribute values when the device was last configured, or before modifying any of its attributes by using the **chdev** command with the **-P** or **-T** flag.

The **-O** flag displays the names of all the attributes that are specified, separated by colons. On the next line, the **-O** flag displays all of 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 by using the **-F** *Format* flag, where the *Format* parameter is a quoted list of column names, separated by non-alphanumeric characters or white space. If the **-F** *Format* flag is specified, the **-Z** *Character* flag can also be specified to change the default record separator from a `newline` character to the indicated *Character*.

The **lsattr** command can display *"operation"* information from the Extended Predefined Attribute (**PdAtXtd**) object class. 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 by using the specified **-f** *File* flag.

Flags

Item	Description
-a *Attribute*	Displays information for the specified attributes of a specific device or type 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 quotation marks with spaces between the names. If you use 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 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 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 it is not used with the **-O** flag. The **-D** flag displays only the attribute name and default value in colon format when it is 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 it is not used with the **-O** flag. The **-E** flag displays only the attribute name and current value in colon format when it is used with the **-O** flag. This flag cannot be used with the **-c**, **-D**, **-F**, **-R**, **-s**, or **-t** flag.
-f *File*	Reads the necessary flags from the *File* parameter.

Item	Description
-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. If white space is used 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 determine whether 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. The **-O** and **-R** flags take precedence over the **-H** flag.
-h	Displays the command usage message.
-l *Name*	Specifies the device logical name in the Customized Devices object class whose attribute names or values you want displayed.
-o *Operation*	Displays information for the specified operations of a specific device or type 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 operation names must be enclosed in quotation marks with spaces between the names. Wildcard characters can also be used for the operation name. The valid set of wildcard characters is the same set that is used by the **odmget** command. All operations that are associated with a specific device, or type of device, can be displayed by using an operation value of `"?*"`. The **-o** *Operation* flag cannot be used 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 specified and default values when the **-D** flag is specified. This flag cannot be used with the **-F** and **-R** flags.
-P	Displays the attribute names, values, descriptions, and user-settable flag values for a specific device when it is not used with the **-O** flag. The values that are displayed are those values with which the device was configured, before any of the device attributes were modified by using the **chdev** command with the **-P** or **-T** flag. When the **-P** flag is used with the **-O** flag, the **-P** flag displays only the attribute name and value in colon-separated format. 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 **-D**, **-E**, **-O**, or **-R** flag.

-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 used with programs that must deal with ODM fields that might have embedded new line characters. The **-Z** *Character* flag is used to change the record separator character for each record, or line, of output generated. The new record separator is specified by using 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 **-R** flag.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current attribute values for the rmt0 tape device, enter the following command:

   ```
   lsattr -l rmt0 -E
   ```

   The system displays a message similar to the following output:

   ```
   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 rmt0 tape device, enter the following command:

   ```
   lsattr -l rmt0 -D
   ```

   The system displays a message similar to the following output:

   ```
   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 scsi0 SCSI adapter, enter the following command:

   ```
   lsattr -l scsi0 -a bus_intr_lvl -E
   ```

   The system displays a message similar to the following output:

   ```
   bus_intr_lvl 1 Bus interrupt level False
   ```

4. To list the possible values of the login attribute for the tty0 tty device, enter the following command:

   ```
   lsattr -l tty0 -a login -R
   ```

   The system displays a message similar to the following output:

   ```
   enable
   disable
   share
   delay
   ```

5. To list the default attribute values for an IBM 4340 parallel printer, enter the following command:

   ```
   lsattr -c printer -s parallel -t ibm4340 -D
   ```

   The system displays a message similar to the following output:

   ```
   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, enter the following command:

   ```
   lsattr -c printer -s parallel -t ibm4340  -a ptop -R
   ```

   The system displays a message similar to the following output:

   ```
   1...1000 (+1)
   ```

7. To list the current attribute values for the rmt0 tape device in colon-separated format, enter the following command:

   ```
   lsattr -l rmt0 -E -O
   ```

   The system displays a message similar to the following output:

   ```
   #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, enter the following command:

   ```
   lsattr -E -l sys0
   ```

   The system displays output similar to the following output:

   ```
   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       256           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 by using SMIT. Select the **System Environments** -> **Change / Show Characteristics of Operating Systems** options to view this information.

Files

Item	Description
/usr/sbin/lsattr	Contains the **lsattr** command.

Related reference:
"lsconn Command" on page 390
"lsdev Command" on page 396
Related information:
chdev command
Tunable parameters

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 augments 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 the *Administering RSCT* 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:

Field	Description
Time	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	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	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	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	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, 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, **#010523042004** corresponds to January 5, 11:04 PM, 2004. 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 the *Administering RSCT* 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, you 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, you 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_IP_AUTHENT

When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Examples

1. To list all records in the audit log on every node in the current management scope as determined by the CT_MANAGEMENT_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_MANAGEMENT_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
   ```

4. To list the records that are generated by the event-response resource manager (ERRM), enter:

   ```
   lsaudrec -SERRM
   ```

5. To list the records that are related to a condition called **Condition1**, enter:

   ```
   lsaudrec -SERRM -s"ConditionName=='Condition1'"
   ```

6. To list the records that are related to an event from **Condition1**, enter:

```
lsaudrec -SERRM -s"ConditionName=='Condition1' && Etype==91"
```

7. To list the records that are related to a rearm event from **Condition1**, enter:

```
lsaudrec -SERRM -s"ConditionName=='Condition1' && Etype==92"
```

8. To list the sensor resource manager records in the audit log on the local node, enter:

```
lsaudrec -SSSRM
```

The output will look like this:

```
Time                Subsystem   Category    Description
11/10/05 21:52:32   SSRM        Error       The Command /SENSOR/sensor.ksh 1
                                            in Sensor SENSOR_NOUSER_1 execution fails.
11/10/05 21:52:36   SSRM        Error       The Command /SENSOR/sensor.nocmd 1 in Sensor
                                            SENSOR_NOCMD_1 exits with error 127.
```

9. To list, in long format, the sensor resource manager records in the audit log on the local node, enter:

```
lsaudrec -l -SSSRM
```

The output will look like this:

```
Time        = 11/10/05 21:52:32 243097
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.ksh 1 in Sensor SENSOR_NOUSER_1 execution fails.
ErrorMsg    = 2645-202 The user name "guest" that was specified for running the command does not exist.

Time        = 11/10/05 21:52:36 361726
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.nocmd 1 in Sensor SENSOR_NOCMD_1 exits with error 127.
StandardOut =
StandardErr = ksh: /u/diane/drmc/scripts/SENSOR/sensor.nocmd:  not found
```

10. To list error records only, enter:

```
lsaudrec -s"Category=1"
```

Location

/opt/rsct/bin/lsaudrec

---

lsauth Command

Purpose

Displays user and system-defined authorization attributes.

Syntax

lsauth [**-R** *load_module*] [**-C**] [**-f**] [**-a** *List*] {**ALL** | *Name* [,*Name*] ...}

Description

The **lsauth** command displays attributes of user-defined and system-defined authorizations from the authorization database. The command can be used to list attributes of all authorizations or specific authorizations. By default, the **lsauth** command displays all authorization attributes. To view selected attributes, use the **-a** *List* flag. If one or more attributes cannot be read, the **lsauth** command lists the information that is available.

By default, the **lsauth** command lists the attributes of each authorization on one line. It displays attribute information in the form of *Attribute* **=** *Value*, each separated by a blank space. To list the authorization attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

If the system is configured to use multiple domains for the authorization database, the authorizations, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the authorizations stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the authorizations from a specific domain.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* parameter requires a blank space between attributes to list multiple attributes. If you specify an empty list, only the authorization names are displayed. The *List* parameter can include any attribute defined in the **chauth** command, in addition to the following two attributes: **description** The text description of the authorization as indicated by the **dfltmsg**, **msgcat**, **msgset** and **msgnum** attributes for the authorization. **roles** A comma-separated list of roles containing the specified authorization in their authorization set.
-C	Displays the authorization attributes in colon-separated records, as follows: *#authorization*:*attribute1*:*attribute2*: ... *authorization*:*value1*:*value2*: ... *authorization2*:*value1*:*value2*: ... The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If an authorization does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.
-f	Displays the output in stanzas, with each stanza identified by an authorization name. Each *Attribute* **=** *Value* pair is listed on a separate line: *Authorization*: *attribute1=value* *attribute2=value* *attribute3=value*
-R *load_module*	Specifies the loadable module to list authorizations from.

Parameters

Item	Description
ALL	Specifies to list attributes from all authorizations.
Name	Specifies the authorization name to list. Optionally, a wild card (.*) can be used at the end of a name to list an entire hierarchy. The entire string specified before the wild card must be a valid authorization name.

Security

The **lsauth** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.auth.list	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/authorizations	r

Examples

1. To display all attributes of the `custom` authorization, use the following command:

   ```
   lsauth custom
   ```

   All the attribute information appears, with each attribute separated by a blank space.
2. To display all attributes of the `custom` authorization from LDAP, use the following command:

   ```
   lsauth -R LDAP custom
   ```

   All the attribute information appears, with each attribute separated by a blank space.
3. To display the authorization ID and description for the `custom` authorization in stanza format, use the following command:

   ```
   lsauth -f -a id description custom
   ```

   Information similar to the following appears:

   ```
   custom:
       id=11000
       description="Custom Authorization"
   ```
4. To display the **msgcat**, **msgset** and **msgnum** attributes for the `custom.test` authorization in a colon format, use the following command:

   ```
   lsauth -C -a msgcat msgset msgnum custom.test
   ```

   Information similar to the following example appears:

   ```
   #name:ID:msgcat:msgset:msgnum
   custom.test:12000:custom_auths.cat:5:24
   ```
5. To display the description for the entire authorization hierarchy that begins with `aix.security`, use the following command:

   ```
   lsauth -a description aix.security.*
   ```

   The `aix.security` authorization and all its children are listed with one authorization per line and a space between the authorization name and the description attribute.

Related information:

ckauth command

getauthattr command

/etc/security/authorizations file

RBAC command

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:

rcp command

get_auth_method command

Communications and networks

Authentication and the secure rcmds

---

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

Item	Description
/usr/sbin/lsC2admin	Contains the lsC2admin 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

Item	Description
/usr/sbin/lsCCadmin	Contains the lsCCadmin 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.

Item	Description
-l *Name*	Displays device information for the named device.
-p	Displays the platform-specific device information.
-v	Displays the VPD found in the Customized VPD object class. Also, 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.


+ indicates the resource has been added to the Diagnostic Resource List. The resource is
added to the Diagnostic Resource list by default when the diagnostic fileset is installed. A
resource must be in the Diagnostic Resource List before diagnostics tasks can be
performed on the resource.

- indicates the resource was deleted from the Diagnostic Resource List.

The resource can be added or deleted from the Resource List by running the diag
command, and using the Task Selection menu to select either Add Resource to Resource
List, or Delete Resource from Resource List.

Diagnostic support for a resource, indicated by the + character is not necessarily
inclusive of all diagnostic tasks. Some resources are only supported with a subset of
diagnostic task, and that subset might or might not include the Run Diagnostics Task.

  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

+ indicates the resource has been added to the Diagnostic Resource List. The resource is
added to the Diagnostic Resource list by default when the diagnostic fileset is installed. A
resource must be in the Diagnostic Resource List before diagnostics tasks can be
performed on the resource.

- indicates the resource was deleted from the Diagnostic Resource List.

The resource can be added or deleted from the Resource List by running the diag
command, and using the Task Selection menu to select either Add Resource to Resource
List, or Delete Resource from Resource List.

Diagnostic support for a resource, indicated by the + character is not necessarily
inclusive of all diagnostic tasks. Some resources are only supported with a subset of
diagnostic task, and that subset might or might not include the Run Diagnostics Task.

+ sys0
        System Object
+ sysplanar0
        System Planar
+ mem0
        Memory
+ L2cache0
        L2 Cache
+ proc0            U5734.100.1234567-P1-C1
        Processor
+ proc1            U5734.100.1234567-P1-C2
        Processor
+hdisk0            U5734.100.1234567-P1-D9
      16 Bit LVD SCSI Disk Drive (4500 MB)
+fd0               U5734.100.1234567-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

Item	Description
/usr/sbin/lscfg	Contains the **lscfg** command.

Related reference:

"lsattr Command" on page 354

"lsconn Command" on page 390

"lsdev Command" on page 396

"lsparent Command" on page 472

lscifscred Command

Purpose

Lists the server or user entries stored in the **/etc/cifs_fs/cifscred** file.

Syntax

lscifscred [**-h** *RemoteHost*] [**-u** *user*]

Description

The **lscifscred** command lists all of the server or user entries that have passwords stored in the **/etc/cifs_fs/cifscred** file.

Flags

Item	Description
-h *RemoteHost*	Lists credentials matching the given remote host (CIFS server) only.
-u *user*	Lists credentials matching the given user name only.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To list all server and user entries on a server named `server1`, enter:
   ```
   lscifscred -h server1
   ```

Location

/usr/sbin/lscifscred

Related information:

chcifscred command

chcifsmnt command

rmcifscred command

rmcifsmnt command

---

lscifsmnt Command

Purpose

Lists the CIFS mounts defined in the **/etc/filesystems** file.

Syntax

lscifsmnt [**-c** | **-l** | **-p**] [*FileSystem*]

Description

The **lscifsmnt** command lists the specified CIFS mounts that are defined in the **/etc/filesystems** file.

Flags

Item	Description
-c	Specifies that the CIFS mount be listed in colon delimited format.
-l	Specifies that the CIFS mount be listed in standard format with each field separated by whitespace. This is the default.
-p	Specifies that the CIFS mount be listed in pipe delimited format.

Parameters

Item	Description
FileSystem	Specifies which file system to list the characteristics of. The default is to list all CIFS file systems.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To list all CIFS mounts defined in **/etc/filesystems**, enter:
   ```
   lscifsmnt
   ```

Location

/usr/sbin/lscifsmnt

Related information:

chcifscred command

chcifsmnt command

rmcifscred command

rmcifsmnt 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

Item	Description

-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:`
	`CPUsoftmax:CPUhardmax:memoryshares:memorymin:`
	`memorysoftmax:memoryhardmax:diskIOshares:diskIOmin:`
	`diskIOsoftmax:diskIOhardmax:totalCPUhardmax:`
	`totalCPUunit:totalDiskIOhardmax:totalDiskIOunit:`
	`totalConnecttimehardmax:totalConnecttimeunit:`
	`totalProcesseshardmax:totalThreadshardmax:`
	`totalLoginshardmax:`
	`classRealMem:classRealMemunit:classVirtMem:`
	`classVirtMemunit:classLargePages:classLargePagesunit:`
	`procVirtMem:procVirtMemunit:localshm:vmenforce:delshm`
	`myclass::0:no::::::-:0:100:100:-:1:100:100:-:0:100:`
	`100:-:s:-:KB:-:s:-:-:-:-:-:KB:-:KB:-:KB:-:KB:no:proc: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:`
	`CPUsoftmax:CPUhardmax:memoryshares:memorymin:`
	`memorysoftmax:memoryhardmax:diskIOshares:diskIOmin:`
	`diskIOsoftmax:diskIOhardmax:totalCPUhardmax:`
	`totalCPUunit:totalDiskIOhardmax:totalDiskIOunit:`
	`totalConnecttimehardmax:totalConnecttimeunit:`
	`totalProcesseshardmax:totalThreadshardmax:totalLoginshardmax:`
	`classRealMem:classRealMemunit:classVirtMem:`
	`classVirtMemunit:classLargePages:classLargePagesunit:`
	`procVirtMem:procVirtMemunit:localshm:vmenforce:delshm`
	`::0:no::::::-:0:100:100:-:0:100:100:-:0:100:100:-:s:-:`
	`KB:-:s:-:-:-:-:-:KB:-:KB:-:KB:-:KB:no:proc: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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
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 reference:
"mkclass Command" on page 679
Related information:
wlmcntrl command
chclass command
rmclass command

lscluster Command

Purpose

Lists the cluster configuration information.

Syntax

lscluster { **-i** | **-d** | **-c** [**-n** *clustername*] } | { **-m** [*nodename*] | **-s** | **-i** *interfacename* | **-d** *diskname* }

Description

The **lscluster** command displays the attributes associated with the cluster and the cluster configuration.

Flags

Item	Description
-c	Lists the cluster configuration.
-d	Lists the cluster storage interfaces.
-i	Lists the network device driver (NDD) and pseudo NDD interfaces that are currently configured on each of the Cluster Aware AIX (CAA) nodes. CAA might not use all of the interfaces to exchange heartbeat packets. **Note:** The storage framework communication (`sfwcom`) interface is displayed as `UP` only if this interface is configured and available. Otherwise, it is not displayed.
-m	Lists the cluster node configuration information. This information includes a list of points of contact. *Points of contact* are cluster configuration interfaces that are used by the cluster to exchange heartbeat packets. If a point of contact has no CAA traffic for an extended period, it is removed from the list of points of contact.
-n	Allows the cluster names to be queried for all interfaces, storage, or cluster configurations (applicable only with **-i**, **-d**, or **-c** flags).
-s	Lists the cluster network statistics on the local node.

Examples

1. To list the cluster configuration for all nodes, enter:

   ```
   lscluster –m
   ```

   The sample of the output follows:

   ```
   # lscluster -m
   Calling node query for all nodes...
   Node query number of nodes examined: 2

                       Node name: nodeA.ibm.com
                       Cluster shorthand id for node: 1
                       uuid for node: 84088524-b124-11e3-8210-32c8e74b1e02
                       State of node:  UP  NODE_LOCAL
                       Smoothed rtt to node: 0
                       Mean Deviation in network rtt to node: 0
                       Number of clusters node is a member in: 1
   ```

```
        CLUSTER NAME       TYPE  SHID   UUID
        Sample local        84ee37f4-b124-11e3-8210-32c8e74b1e02

        Number of points_of_contact for node: 0
        Point-of-contact interface & contact state
         n/a

------------------------------

        Node name: nodeB.ibm.com
        Cluster shorthand id for node: 2
        uuid for node: 8492a5a6-b124-11e3-8210-32c8e74b1e02
        State of node:  UP
        Smoothed rtt to node: 70
        Mean Deviation in network rtt to node: 82
        Number of clusters node is a member in: 1
        CLUSTER NAME       TYPE  SHID   UUID
        Sample local        84ee37f4-b124-11e3-8210-32c8e74b1e02

        Number of points_of_contact for node: 2
        Point-of-contact interface & contact state
        dpcom  UP  RESTRICTED
        en0  UP
```

2. To list the cluster configuration for the local node, enter:

   `lscluster -s`

   The sample of the output follows:

```
# lscluster -s
        Cluster Network Statistics:

        pkts seen: 33861217                  passed: 32052241
        IP pkts: 5778096                     UDP pkts: 1934943
        gossip pkts sent: 1463320            gossip pkts recv: 688759
        cluster address pkts: 0              CP pkts: 1808962
        bad transmits: 5                     bad posts: 4
        Bad transmit (overflow - disk ): 0
        Bad transmit (overflow - tcpsock): 0
        Bad transmit (host unreachable): 0
        Bad transmit (net unreachable): 0
        Bad transmit (network down): 0
        Bad transmit (no connection): 0
        short pkts: 0                        multicast pkts: 1808880
        cluster wide errors: 0               bad pkts: 0
        dup pkts: 0                          dropped pkts: 14
        pkt fragments: 1                     fragments queued: 0
        fragments freed: 0
        pkts pulled: 0                       no memory: 0
        rxmit requests recv: 10              requests found: 3
        requests missed: 7                   ooo pkts: 0
        requests reset sent: 7               reset recv: 0
        remote tcpsock send: 0               tcpsock recv: 0
        rxmit requests sent: 0
        alive pkts sent: 0                   alive pkts recv: 0
        ahafs pkts sent: 2                   ahafs pkts recv: 0
        nodedown pkts sent: 0                nodedown pkts recv: 1
        socket pkts sent: 62                 socket pkts recv: 54
        cwide pkts sent: 275321              cwide pkts recv: 275318
        socket pkts no space: 0              pkts recv notforhere: 0
        Pseudo socket pkts sent: 0           Pseudo socket pkts recv: 0
        Pseudo socket pkts dropped: 0
        arp pkts sent: 1                     arp pkts recv: 2
        stale pkts recv: 0                   other cluster pkts: 4
        storage pkts sent: 1                 storage pkts recv: 1
        disk pkts sent: 174                  disk pkts recv: 0
        unicast pkts sent: 275364            unicast pkts recv: 82
        out-of-range pkts recv: 0
```

```
            IPv6 pkts sent: 0                   IPv6 pkts recv: 122
            IPv6 frags sent: 0                  IPv6 frags recv: 0
            Unhandled large pkts: 0
            mrxmit overflow     : 0             urxmit overflow: 0
```

3. To list the interface information for the local node, enter:

   ```
   lscluster –i
   ```

   The sample of output follows:

```
# lscluster -i
            Network/Storage Interface Query

            Cluster Name:  Sample
            Cluster uuid:  84ee37f4-b124-11e3-8210-32c8e74b1e02
            Number of nodes reporting = 2
            Number of nodes expected = 2

            Node nodeA.ibm.com
            Node uuid = 84088524-b124-11e3-8210-32c8e74b1e02
            Number of interfaces discovered = 2
                    Interface number 1 en0
                            ifnet type = 6 ndd type = 7
                            Mac address length = 6
                            Mac address =  32:C8:E7:4B:1E:02
                            Smoothed rrt across interface = 0
                            Mean Deviation in network rrt across interface = 0
                            Probe interval for interface = 100 ms
                            ifnet flags for interface = 0x1E080863
                            ndd flags for interface = 0x0021081B
                            Interface state  UP
                            Number of regular addresses configured on interface = 1
                            IPv4 ADDRESS: 9.3.199.216  broadcast 9.3.199.255  netmask 255.255.254.0
                            Number of cluster multicast addresses configured on interface = 1
                            IPv4 MULTICAST ADDRESS: 228.3.199.216  broadcast 0.0.0.0  netmask 0.0.0.0
                    Interface number 2 dpcom
                            ifnet type = 0 ndd type = 305
                            Mac address length = 0
                            Mac address =  00:00:00:00:00:00
                            Smoothed rrt across interface = 750
                            Mean Deviation in network rrt across interface = 1500
                            Probe interval for interface = 22500 ms
                            ifnet flags for interface = 0x00000000
                            ndd flags for interface = 0x00000009
                            Interface state  UP  RESTRICTED  AIX_CONTROLLED
                    Pseudo Interface
                            Interface State DOWN

            Node nodeB.ibm.com
            Node uuid = 8492a5a6-b124-11e3-8210-32c8e74b1e02
            Number of interfaces discovered = 2
                    Interface number 1 en0
                            ifnet type = 6 ndd type = 7
                            Mac address length = 6
                            Mac address =  32:C8:EF:AD:7C:02
                            Smoothed rrt across interface = 0
                            Mean Deviation in network rrt across interface = 0
                            Probe interval for interface = 990 ms
                            ifnet flags for interface = 0x1E084863
                            ndd flags for interface = 0x0021081B
                            Interface state  UP
                            Number of regular addresses configured on interface = 1
                            IPv4 ADDRESS: 9.3.199.128  broadcast 9.3.199.255  netmask 255.255.254.0
                            Number of cluster multicast addresses configured on interface = 1
                            IPv4 MULTICAST ADDRESS: 228.3.199.216  broadcast 0.0.0.0  netmask 0.0.0.0
                    Interface number 2 dpcom
                            ifnet type = 0 ndd type = 305
                            Mac address length = 0
```

```
                        Mac address =  00:00:00:00:00:00
                        Smoothed rrt across interface = 750
                        Mean Deviation in network rrt across interface = 1500
                        Probe interval for interface = 22500 ms
                        ifnet flags for interface = 0x00000000
                        ndd flags for interface = 0x00000009
                        Interface state  UP  RESTRICTED  AIX_CONTROLLED
                Pseudo Interface
                        Interface State DOWN
```

4. To list the storage interface information for the cluster, enter:

   ```
   lscluster -d
   ```

   The sample of output follows:

   ```
   # lscluster -d
               Storage Interface Query

               Cluster Name:  Sample
               Cluster uuid:  84ee37f4-b124-11e3-8210-32c8e74b1e02
               Number of nodes reporting = 2
               Number of nodes expected = 2
               Node nodeA.ibm.com
               Node uuid = 84088524-b124-11e3-8210-32c8e74b1e02
               Number of disk discovered = 1
                       hdisk4
                         state : UP
                         uDid  :
                         uUid  : 76c94719-7335-ded6-10e2-77d61ff7998c
                         type  : REPDISK
               Node nodeB.ibm.com
               Node uuid = 8492a5a6-b124-11e3-8210-32c8e74b1e02
               Number of disk discovered = 1
                       hdisk0
                         state : UP
                         uDid  : 382300c4f4f700004c0000000140799c6e39.3105VDASD03AIXvscsi
                         uUid  : 76c94719-7335-ded6-10e2-77d61ff7998c
                         type  : REPDISK
   ```

5. To list the cluster configuration, enter:

   ```
   lscluster -c
   ```

   The sample of the output follows:

   ```
   # lscluster -c
   Cluster Name: Sample
   Cluste UUID: 8e1d89da-b39d-11e3-91e7-d24dc2d9d309
   Number of nodes in cluster = 2
           Cluster ID for node nodeA.ibm.com: 1
           Primary IP address for node r5r3m25.aus.stglabs.ibm.com: 9.3.207.132
           Cluster ID for node nodeB.ibm.com: 2
           Primary IP address for node r5r3m26.aus.stglabs.ibm.com: 9.3.207.218
   Number of disks in cluster = 1
           Disk = hdisk6 UUID = 57208624-fda4-d404-a7c0-8e425e2941a4 cluster_major = 0 cluster_minor = 1
   Multicast for site LOCAL: IPv4 228.3.207.132 IPv6 ff05::e403:cf84
   Communication Mode: multicast
   Local node maximum capabilities: HNAME_CHG, UNICAST, IPV6, SITE
   Effective cluster-wide capabilities: HNAME_CHG, UNICAST, IPV6, SI
   ```

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:

Field	Description
Name	The name of the communication group
Sensitivity	The number of missed heartbeats that constitute a failure
Period	The number of seconds between heartbeats
Priority	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	The path to the Network Interface Module (NIM) that supports the adapter types in the communication group
NIMParameters	The NIM start parameters

Interface resources

Use the **-i** flag to display information about the interface resources that refer to *communication_group*.

For IP communication groups (**MediaType** = **1**), **lscomg -i** displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.
NodeName	The host name of the interface resource that refers to *communication_group*.
IPAddress	The IP address of the interface resource that refers to *communication_group*.
SubnetMask	The subnet mask of the interface resource that refers to *communication_group*.
Subnet	The subnet of the interface resource that refers to *communication_group*

For disk heartbeating (**MediaType** = **2**) and other non-IP types of communication groups (**MediaType** = **0**), **lscomg -i** displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.
NodeName	The host name of the interface resource that refers to *communication_group*.
DeviceInfo	Information about the device.
MediaType	The type of interfaces that make up this 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 the name of the communication group about which you want to display information. 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.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain on which the communication group exists.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** is 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 following output is displayed:

   ```
   Name  Sensitivity  Period  Priority  Broadcast  SourceRouting  NIMPath                     NIMParameters
   ComG1      2          2        1        no          yes        /opt/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 following output is displayed:

   ```
   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

/opt/rsct/bin/lscomg

---

lscondition Command

Purpose

Lists information about one or more conditions.

Syntax

lscondition [**-a**] [**-m** | **-n** | **-e**] [**-C** | **-l** | **-t** | **-d** | **-D** *delimiter*] [**-A**] [**-q**] [**-U**] [**-x**] [**-h**] [**-TV**] [*condition1* [**,***condition2*,...]**:***node_name*]

Description

The **lscondition** command lists the following information about defined conditions:

Field	Description
Name	The name of the condition.
Node	The location of the condition (for management domain scope or peer domain scope).
MonitorStatus	The status of the condition.
ResourceClass	The resource class that is monitored by this condition.
EventExpression	The expression that is used in monitoring this condition.
EventDescription	A description of the **EventExpression** field.
RearmExpression	The expression used in determining when monitoring should restart for this condition after an event has occurred.
RearmDescription	A description of the **RearmExpression** field.
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.
NodeNames	The host names of the nodes where the condition is registered.
MgtScope	The RMC scope in which the condition is monitored.
Toggle	Specifies whether the condition toggles between the event and the rearm event.
Locked	Specifies whether the resource is locked or unlocked.
EventBatchingInterval	Specifies the time in seconds that is used to determine when the accumulated events are batched together and sent to the response. A value of 0 indicates that no batching is used.
EventBatchingMaxEvents	Specifies the maximum number of events that can be in a single batch of events. A value of 0 indicates that there is no maximum if the value of **EventBatchingInterval** is not 0.
BatchedEventRetentionPeriod	Specifies the time in hours that the batched event file is kept after all associated response scripts are run.
BatchedEventMaxTotalSize	Specifies that the total saved batched event file size can't exceed a certain size in megabytes (MB) per condition. RecordAuditLog Specifies the level of detail for ERRM log entries to the audit log (ALL, Error Only, or None).

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.

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.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node.

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.

-A Displays all of the attributes of the condition.

-C Displays a **mkcondition** command template based on the condition. By modifying this template, you can create new conditions. If more than one condition is specified, the template for each **mkcondition** command appears on a separate line. This flag is ignored when no conditions are specified. This flag overrides the **–l** flag.

-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.

-e Lists only those conditions that are monitored in error.

-l Produces long-formatted output. Displays the condition information on separate lines.

-m Lists only those conditions that are being monitored without error.

-n Lists only those conditions that are not being monitored.

-q Does not return an error when the condition does not exist.

-t Displays the condition information in separate columns (table format).

-U Indicates whether the resource is locked.

-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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To list all conditions and their monitoring status, run this command:

   ```
   lscondition
   ```

   The output will look like this:

   ```
    Name                       Node          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"
   Node             = "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"
   NodeNames        = "{}"
   MgtScope         = "l"
   Toggle           = "Yes"
   Locked           = "No"
   ```

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, 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 -a
   ```

   The output will look like this:

   ```
   Name                       Node        MonitorStatus
   "FileSystem space used"    "nodeA"     "Monitored"
   "tmp space used"           "nodeB"     "Not monitored"
   "var space used"           "nodeC"     "Error"
   ```

Location

/opt/rsct/bin/lscondition

---

lscondresp Command

Purpose

Lists information about a condition and any of its condition/response associations.

Syntax

To list the link between a condition and one or more responses:

lscondresp [**-a** | **-n**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-q**] [**-U**] [**-x**] [**-z**] [**-h**] [**-TV**] [*condition*[**:***node_name*] [*response1* [*response2*...]]]

To list all of the links to one or more responses:

lscondresp [**-a** | **-n**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-q**] [**-x**] [**-z**] **-r** [**-U**] [**-h**] [**-TV**] *response1*[**:***node_name*] [*response2*...]

Description

The **lscondresp** command lists information about a condition and its linked responses. A link between a condition and a response is called a *condition/response association.* 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:

Field	Description
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.
Node	The location of the condition and the response.
Locked	Indicates whether the resource is locked or unlocked.

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 condition/response associations that are defined on that node. List all of the condition/response associations 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 condition/response associations are listed. For local scope, only condition/response associations on the local node are listed. For management domain scope and peer domain scope, the condition/response associations 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.

-U Indicates whether the resource is locked.

-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 Lists information about all of the condition/response associations for the specified responses. Use this flag to indicate that all command parameters specified are responses, not conditions.

-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 or 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.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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in

processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

To see which resources are locked, run this command:

```
lscondresp -U
```

The output will look like this:

```
Condition             Response                 Node     State         Locked
"/tmp space used"     "E-mail root off-shift"  "nodeA"  "Not active"  "Yes"
"Page space in rate"  "E-mail root anytime"    "nodeA"  "Not active"  "No"
```

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                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   "Page in Rate"           "Log event anytime"           "nodeA"    "Active"
   ```

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                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   ```

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                       Node       State
"FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
```

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                       Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   ```

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                       Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   ```

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                       Node       State
   "FileSystem space used"   "Broadcast event on-shift"    "nodeB"    "Active"
   "FileSystem space used"   "E-mail root anytime"         "nodeB"    "Not Active"
   ```

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                       Node     State
   "FileSystem space used"   "Broadcast event on-shift"    "nodeC"    "Active"
   "FileSystem space used"   "E-mail root anytime"         "nodeC"    "Not Active"
   ```

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                       Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeD"    "Active"
   "FileSystem space used"  "E-mail root anytime"         "nodeD"    "Not Active"
   "FileSystem space used"  "Broadcast event on-shift"    "nodeE"    "Active"
   "FileSystem space used"  "E-mail root anytime"         "nodeE"    "Not Active"
   ```

Location

/opt/rsct/bin/lscondresp

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 about 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.

Use the flags either on the command line or in the specified **-f** *File* flag.

Flags

Item	Description
-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 necessary 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 non-alphanumeric characters or white space. If white space id used 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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all of the possible connection locations on the sa2 IBM 8-Port EIA-232/RS-422A (PCI) Adapter that will accept an RS-232 device connection, type the following:

   ```
   lsconn -p sa2 -k rs232
   ```

   The system displays a possible connections similar to the following:

   ```
   0
   1
   2
   3
   4
   5
   6
   7
   ```

2. To list all of the possible connection locations and connection types on the sa2 IBM 8-Port EIA-232/RS-422A (PCI) Adapter, type the following:

   ```
   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

Item	Description
/usr/sbin/lsconn	Specifies the command file.

Related reference:

"lsdev Command" on page 396

"lsparent Command" on page 472

"mkdev Command" on page 701

Related information:

chdev command

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

Item	Description
-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. **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:

Item	Description
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

Item	Description
/usr/sbin/lscons	Contains the **lscons** command.

Related information:

chcons command

swcons command

console special file

---

lscore Command

Purpose

Views the current core settings.

Syntax

lscore [**-R** *registry*] [*username* | **-d**]

Description

The **lscore** command will be the user interface to view the current core settings. It will have the following usage:

```
lscore [-R registry] [username|-d]
```

As with **chcore**, the **-d** flag will show the default values. Viewing settings for another user is a privileged operation; however, any user may view the default values.

Flags

Item	Description
-d	Changes the default setting for the system.
-R *registry*	Specifies the loadable I&A module.

Security

May only be run by root or another user with system authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the current settings for root, type:

   ```
   lscore root
   ```

   The output will look like:

   ```
   compression: on
   path specification: default
   corefile location: default
   naming specification: off
   ```

2. To list the default settings for the system, type:

   ```
   lscore -d
   ```

The output will look like:

```
compression: off
path specification: on
corefile location: /corefiles
naming specification: off
```

Related information:

chcore command

lscosi Command

Purpose

Lists information related to a Common Operating System Image (COSI).

Syntax

lscosi [[**-l**{**1**|**2**|**3**}] . . .] [**-v**] [*COSI*]

Description

The **lscosi** command lists the status and detailed information related to a Common Operating System Image (COSI). The level of information to be listed depends on the numeric value specified by the **-l** flag, with a level ranging from 1 - 3 (3 being the most detailed). If a level is not specified, a default of level 1 information is displayed. If no argument is specified, the **lscosi** command lists any common images that exist in the environment. The **bos.sysmgt.nim.master** fileset must be present on the system in order for the **lscosi** command to be successful. This command can also be executed on a thin server.

Flags

Item	Description
-l{**1**|**2**|**3**}	Specifies the level of information to display. **1** This level displays very limited information related to a COSI. The information listed shows only a brief summary of the COSI and the thin servers that might be using it. **2** This level displays more than just basic information related to a COSI. The level includes information pertaining to the software content of the COSI. **3** This level displays more in-depth information related to a COSI. The level includes information pertaining to the installation log of the COSI.
-v	Enables verbose debug output when the **lscosi** command runs.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **lscosi** command.

Examples

1. To check if any common images exist in an environment, enter:

   ```
   lscosi
   ```

   When this command is entered without an argument, it merely lists common images in the environment. The output might be similar to the following list:

   ```
   52H_0442A_cosi
   52I_0444B2_GOLD_cosi
   52L_0534A_cosi
   53E_0545A_cosi
   53D_GOLD_cosi
   53A_GOLD_cosi
   52M_0544A_cosi
   ```

2. To list brief status information for a common image named `cosi1`, enter:

   ```
   lscosi cosi1
   ```

   Information similar to the following is displayed:

   ```
   53H_0538A_spot:
     class         = resources
     type          = spot
     plat_defined  = chrp
     Rstate        = ready for use
     prev_state    = verification is being performed
     location      = /export/nimvg/spot/53H_0538A_spot/usr
     version       = 5
     release       = 2
     mod           = 0
     oslevel_r     = 5300-05
     alloc_count   = 2
     server        = master
     if_supported  = chrp.mp ent
     Rstate_result = success

   Thin Server:
   Client1
   Client2
   ```

3. To list software content for a common image named `cosi1`, enter:

   ```
   lscosi -l2 cosi1
   ```

   Software content similar to the following is displayed from the common image:

   ```
   Fileset                   Level  State  Type  Description
   (Uninstaller)
   ----------------------------------------------------------
   bos.64bit                5.2.0.75  C     F    Base Operating System 64 bit Runtime
   bos.diag.com             5.2.0.75  C     F    Common Hardware Diagnostics
   bos.diag.rte             5.2.0.75  C     F    Hardware Diagnostics
   .
   .
   .
   ```

4. To list both software content and status information for a common image named `cosi1`, enter:

```
lscosi -l1 -l2 cosi1
```

Location

/usr/sbin/lscosi

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

Related information:
chcosi command
cpcosi command
nim command
nimconfig command

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*] [**-x**]

lsdev -P [**-c** *Class*] [**-s** *Subclass*] [**-t** *Type*] [**-f** *File*] [**-F** *Format* | **-r** *ColumnName*] [**-h**] [**-H**] [**-x**]

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 the following ways:
- From the Customized Devices object class by using the **-C** flag
- From the Predefined Devices object class by using the **-P** flag

To override these two default outputs, use the **-F** *Format* flag to display the output in a format that you specify by using the *Format* parameter. The *Format* parameter is a quoted list of column names that are separated and possibly ended by non-alphanumeric characters or white space.

The **lsdev** command shows information only about devices that are 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 might be conditions where a device might not be displayed. For example, if the **-p** *Parent* flag is used, but the parent that is identified in the Customized Devices object for a device does not match the *Parent* that is specified through the **-p** flag, the device is not displayed. However, the device might have a path to the specified *Parent* that is defined in the Customized Paths object class. Use the **lspath** command to show all MPIO-capable child devices of the specified parent.

You can use the System Management Interface Tool (SMIT) **smit lsdev** fast path to change device characteristics.

Flags

Item	Description
-C	Lists information about a device that is in the Customized Devices object class. The default information that is displayed is *name*, *status*, *location*, and *description*. The **-C** flag is not required, but is maintained for compatibility reasons. The **-C** flag cannot be specified with the **-P** flag. If neither is specified, the **lsdev** command behaves as if the **-C** flag was 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 necessary 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. If white space is used 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 specify only 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 physical location code of the device.
-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 of the device for which information is listed. The *Name* argument to the **-l** flag can contain the same wildcard characters that can be used with the **odmget** command. If the *Name* argument is a dash, names are read from STDIN. Names on STDIN must be separated by a comma, a tab, a space, or a "newline" character. 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 specified *Parent*. The *Parent* argument to the **-p** flag might contain the same wildcard characters 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 that is displayed is *class*, *type*, *subclass*, and *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 of 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 specify only 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. The *State* parameter can have one of the following values: • *d*, *D*, *0* or *defined* for the Defined state • *a*, *A*, *1*, or *available* for the Available state • *s*, *S*, *2*, or *stopped* for the Stopped state This flag can be used to restrict output to devices in a specified state. This flag cannot be used with the **-P** flag.
-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.
-x	Displays the exported status for devices that are exported to a Workload Partition (WPAR).

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list all of the devices in the Predefined Devices object class with column headers, type the following command:

   ```
   lsdev -P -H
   ```

   The system displays a message similar to the following output:

   ```
   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
   .
   .
   .
   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 of the devices in the Customized Devices object class, type the following command:

   ```
   lsdev -C
   ```

   The system displays a message similar to the following output:

   ```
   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 the following command:

   ```
   lsdev -C -c adapter -S a
   ```

   The system displays a message similar to the following output:

   ```
   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 of the tape devices in the Predefined Devices object class, type the following command:

   ```
   lsdev -P -c tape
   ```

   The system displays a message similar to the following output:

   ```
   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 the following command:

   ```
   lsdev -P -r class
   ```

   The system displays a message similar to the following output:

   ```
   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
tty
```

6. To list the supported subclasses in the Predefined Devices object class for the **disk** class, type the following command:

   ```
   lsdev -P -c disk -r subclass
   ```

   The system displays a message similar to the following output:

   ```
   dar
   fcp
   fdar
   ide
   iscsi
   scraid
   scsi
   vscsi
   ```

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 the following command:

   ```
   lsdev -C -H -S a -F 'name class subclass type'
   ```

   The system displays a message similar to the following output:

   ```
   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 in the Customized Devices object class with column headers, type the following command:

   ```
   lsdev -C -c adapter -F 'name class location physloc'
   ```

   The system displays a message similar to the following output:

   ```
   ent0   adapter 02-08 UTMP0.02F.00004BA-P1-C3-T1
   scsi0  adapter 01-08 UTMP0.02F.00004BA-P1-C2-T1
   scsi1  adapter 01-09 UTMP0.02F.00004BA-P1-C2-T2
   scsi2  adapter 03-08 UTMP0.02F.00004BA-P1-C4-T1
   scsi3  adapter 03-09 UTMP0.02F.00004BA-P1-C4-T2
   vsa0   adapter       U9111.520.10004BA-V4-C0
   vscsi0 adapter       U9111.520.10004BA-V4-C2
   vscsi1 adapter       U9111.520.10004BA-V4-C3
   ```

9. To list all of the children of the `pci0` bus, type the following command:

   ```
   lsdev -p pci0
   ```

   The system displays a message similar to the following output:

   ```
   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)
   ```

10. 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 output:

    ```
    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
    ```

11. To display the status of the devices that are exported to a WPAR as **Exported**, enter the following command:

    ```
    # lsdev -c disk -x
    ```

    The system displays a message similar to the following output:

    ```
    hdisk0 Available 01-08-00-1,0 16 Bit LVD SCSI Disk Drive
    hdisk1 Exported  01-08-00-2,0 Other SCSI Disk Drive
    ```

    where the `hdisk1` device is exported to a WPAR.

Files

Item	Description
/usr/sbin/lsdev	Contains the **lsdev** command.

Related reference:

"lsattr Command" on page 354

"mkdev Command" on page 701

Related information:

chdev command

rmdev command

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

Item	Description
-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:

```
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

Item	Description
bin/lsdisp	Contains the **lsdisp** command.

Related information:

chdisp command

Low Function Terminal (LFT) Subsystem Overview

lsdom Command

Purpose

Displays domain attributes.

Syntax

lsdom [-C] [-f] [-a *Attr* [*Attr*]...] { ALL | Name [, *Name*] ...}

Description

The **lsdom** command displays the attributes of the domain, which is defined from the domain database.

The command enables you to list attributes of all domains or specific domains. By default the **lsdom** command displays all domain attributes. To view selected attributes, use the **-a** list flag. If one or more attributes cannot be read, the **lsdom** command lists as much information as possible.

By default, the **lsdom** command lists each domain's attributes in one line. It displays attribute information as *Attribute*=*Value* definitions, each separated by a blank space. To list the domain attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

Flags

Item	Description
-C	Displays the domain attributes in colon-separated records. *#domain:attribute1:attribute2: ...* *domain1:value1:value2: ...* *domain2:value1:value2: ...*
-f	Displays the output in stanzas, with each stanza identified by a domain name. Each *Attribute*=*Value* pair is listed in a separate line. **Domain:** *attribute1=value*

Parameters

Item	Description
ALL	Indicates that the attributes of all domains must be listed.
Name	Indicates the domain name whose attributes must be listed.

Security

The **lsdom** command is a privileged command. Callers of the command must have activated a role that has the following authorization to run the command successfully.

Item	Description
aix.security.domains.list	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/domains	r

Examples

1. To display all attributes of the domain hrdom:

```
lsdom hrdom
```

All the attribute information is shown with each attribute separated by a blank space.

Related information:

mkdom command

setkst command

getdomattr command
/etc/security/domains file
RBAC command

---

lsevent Command

Purpose

Lists event-monitoring information from the audit log.

Syntax

To list events from the audit log:

lsevent [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e a** | **r** | **b**] **[-i]** [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] **[-h]** **[-TV]**

To list responses from the audit log:

lsevent -r [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] **[-i]** [**-a** | **n** *node1*[**,***node2*...]] **[-h]** **[-TV]** [response [response...]]

To list events for a condition from the audit log:

lsevent [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e a** | **r** | **b**] **[-i]** [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] **[-h]** **[-TV]** *condition*

To list responses for a condition from the audit log:

lsevent -R [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] [-i] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [-h] [-TV] *condition* [*response* [*response*...]]

To list events and responses for a condition from the audit log:

lsevent -A [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] [-i] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [-h] [-TV] condition [*response* [*response*...]]

Description

The **lsevent** command lists event-monitoring information from the audit log. The audit log contains information about monitored events or conditions, and responses that were run as a result. This information allows a system administrator to see how events are being processed. The **lsevent** command lists only the information from the audit log recorded by RSCT event response resource manager (ERRM). By using **lsevent**, you can list audit log information without knowing detailed information about ERRM audit log templates, as you would need using the **lsaudrec** command.

By default, without using options and operands, the **lsevent** command lists the events that are recorded in the audit log. These events describe the monitored events that occurred. To list the events for a particular condition, specify the condition name.

Response information can be listed separately or with the event information. Responses are run based on a condition or event occurring. Information about a response includes when it was run, what the response script was, the return code, the expected return code, standard error output, and standard output. To see standard output and the expected return code, the response resource must be defined to

record it by **mkresponse** or **chresponse**. To list only response information, use the **-r** flag. You can optionally specify one or more response names to limit the number of responses listed.

To list event information and response information for a condition, you can use the **-R** and **-A** flags with a condition name. Without **-R** and **-A**, when a condition is specified, the events for the condition are listed. Specify **-R** to list the responses for the condition. You can specify one or more response names to limit the output to those responses. Specify **-A** to list the events and the responses. You can specify one or more response names to limit the response output for **-A** as well. If a condition and at least one response are specified without specifying the **-R**, **-A**, or **-r** flags, **-R** is assumed.

The type of event listed can be controlled using the **-e** flag. You can list events, rearm events, and error events for a condition. The **-w** flag can be used to list events that occurred on a particular node. The **-w** flag has meaning when it is used in listing events. Status information is displayed when the **-i** flag is specified. When listing conditions, the status information includes showing when the condition was registered and unregistered, and when event errors occur. For response information, the status information shows that a response is about to run.

Use the **-B** and **-E** flags if you need to specify a time to limit the command output. By default, **lsevent** lists all audit log entries according to the flags specified, but you can specify a beginning time or an ending time if you are interested in a certain period. The time format is described below. The **-O** flag is used to limit the search of the audit log to the most recent records. The value used with the **-O** flag determines how many of the most recent records are searched for the other **lsevent** criteria specified. For example, using **lsevent -O 1000** causes **lsevent** to search the most recent 1000 records in the audit log for events. If **-a** or **-n** is used, **-O** cannot be used.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Parameters

condition
Specifies the name of a condition for which audit log information is listed.

response
Specifies the name of a response for which audit log information is listed.

Flags

-a Specifies that the **lsevent** command retrieves audit log information from all of the nodes in the cluster. The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to **2**.

-A Specifies that event and response information for a condition is to be listed.

-B *MMddhhmmyyyy*
Specifies to list the audit log entries beginning at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-e a | r | b | e | A
Specifies the type of event to list from the audit log. The following parameters can be specified along with the **-e** flag:

a Lists events from conditions. It is the default setting.

r Lists rearm events from conditions.

b List events and rearm events from conditions.

e Lists response information that is triggered by error events. This setting is meaningful only when **-r**, **-R**, or **-A** is specified.

A Lists all types of events (events, rearm events, and error events).

More than one event type can be specified, for example: **-e ae**.

If the **-e** flag is specified with the **-r** or **-R** flags, the response log entry for the batch-enabled condition is always displayed because the batched events file can contain all type of events.

-E *MMddhhmmyyyy*
Specifies to list the audit log entries up to or ending at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-i Specifies that status information for a condition or response is to be listed. The status information includes information about event registration, event errors, and responses about to be run.

n *node1***[,***node2***...]**
Specifies the node or nodes from which the audit log information is to be retrieved. If node is not specified, the local node is used. *node* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

-O *entries*
Specifies that only the latest entries in the audit log are searched for information. *entries* determines how many of the most recent records are search for the other **lsevent** criteria specified. For example, using **-O 1000** causes the **lsevent** command to search the most recent 1000 records in the audit log for events.

-r Specifies that all command parameters are response names and that response information is to be returned for the responses specified. There are no condition names in the parameter list. If no response names are specified, then information is listed for all responses.

-R Specifies that only the response information for a condition is to be listed.

-w *event_node*
Specifies the node on which the event occurred. This flag is only meaningful in listing events.

-h Writes this command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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 usage statement is written to standard output.

Standard error

All trace messages are written to standard error.

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.

Restrictions

If you are using the **lsevent** command, you must have read access to the ERRM audit log resource 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.

Implementation specifics

This command is part of the **rsct.core** fileset for the AIX operating system and **rsct.core-v.r.m.s-0.***platform***.rpm** package for the Linux, Solaris, and Windows platforms, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/opt/rsct/bin/lsevent

Examples

1. To list the information for events that occurred, enter:
   ```
   lsevent
   ```
2. To list the event information for a condition named **Condition1**, enter:
   ```
   lsevent Condition1
   ```
3. To list the event response information, enter:
   ```
   lsevent -r
   ```
4. To list the event response information for a response named **Response1**, enter:
   ```
   lsevent -r Response1
   ```
5. To view the output of the event response named **Response1**, which is defined to save its output, enter:
   ```
   lsevent -r Response1
   ```
6. To see the events found in the latest 1000 audit log records, enter:
   ```
   lsevent -O 1000
   ```
7. To list the rearm event information for a condition named **Condition1**, enter:
   ```
   lsevent -e r Condition1
   ```

---

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

---

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 the System Management Interface Tool (SMIT) **smit lsfont** fast path to run this command.

Flags

Item	Description
-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
```

Files

Item	Description
/bin/lsfont	Contains the **lsfont** command.
/usr/lpp/fonts	Contains fonts directory.

Related reference:

"mkfont Command" on page 714

Related information:

chfont command

Low Function Terminal (LFT) Subsystem Overview

Text fonts

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 the System Management Interface Tool (SMIT) **smit lsfs** fast path to run this command.

Flags

Item	Description
-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	Displays additional Journaled File System (JFS) or Enhanced Journaled File System (JFS2) characteristics specific to the file system type. 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

Item	Description
/etc/filesystems	Lists the known file systems and defines their characteristics.

Related information:

chfs command

crfs command

File systems

Mounting command

---

lsgroup Command

Purpose

Displays group attributes.

Syntax

lsgroup [**-R** *load_module*] [**-c** | **-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 that are described in the **chgroup** command are displayed. If the **lsgroup** command cannot read one or more attributes, it lists as much information as possible, but does not display empty attributes. To view a selected attribute, use the **-a** *List* flag.

Note: If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, the **lsgroup** command lists the users from the LDAP module and the LOCAL module, if present.

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** or **-C** flag.

You can use the System Management Interface Tool (SMIT) **smit lsgroup** fast path to run this command.

Flags

Item	Description
-a *List*	Specifies the attributes to display. The *List* parameter can include any attribute that is 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: ...`
	If a value contains a **:** symbol, then in the output **:** symbol is prefixed with the **#!** symbols.

Item	Description

-C	Displays the group attributes in colon-separated records that are easier to parse than the output of the **-c** flag:
	`#name:attribute1:attribute2: ...`
	`Group1:value1:value2: ...`
	`Group2:value1:value2: ...`
	
	The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you also specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If you do not have a value for a given attribute, the field is still displayed, but is empty. If a value contains a **:** symbol, then in the output the **:** symbol is prefixed with **#!** symbols. The last field in each entry ends with a newline character rather than a colon.

-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 that is used to get the group attribute list. If the *domainlessgroups* attribute is set in the/etc/secvars.cfg file and the **-R** *LDAP* command is used, then the attribute list is obtained from the LOCAL module, if the group exists on the LOCAL module, and does not exist on the LDAP module. This condition also applies to the **-R** *files* command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

Security

Access Control: This command must 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 might not be able to access all the information. This attribute depends on the access policy of your system. This command must have the *trusted computing base* attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
r	**/etc/group**
r	**/etc/security/group**
r	**/etc/passwd**

Limitations

Listing a group might 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, enter the following command:
   ```
   lsgroup finance
   ```
2. To display the `id`, members (`users`), and administrators (`adms`) of the `finance` group in stanza format, enter the following command:
   ```
   lsgroup  -f  -a id users adms finance
   ```
3. To display the attributes of all the groups in colon-separated format, enter the following command:
   ```
   lsgroup  -c ALL
   ```
   All the attribute information is displayed, with each attribute separated by a blank space.
4. To display the attributes of the LDAP I&A loadable module group monsters, enter the following command:
   ```
   lsgroup -R LDAP monsters
   ```

Files

Item	Description
/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 reference:

"mkgroup Command" on page 719

"mkuser Command" on page 839

Related information:

chgroup command

chgrpmem command

rmuser command

lsiscsi Command

Purpose

Displays information for iSCSI target data.

Syntax

lsiscsi [**-l** *AdapterName*] [**-g** *group*] [**-p**] [**-F** *Format*]

Description

The **lsiscsi** command displays iSCSI target data from ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The second category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the **static** and **auto** groups, respectively, specified by the **-g** flag.

Flags

Item	Description
-F *Format*	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. If white space is used as the separator, the **lsiscsi** command displays the output in aligned columns.
-g *group*	Specifies which group this iSCSI target is associated with. There two valid groups are **static** and **auto**. The **static** group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The **auto** group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
-l *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
-p	Displays the iSCSI target's password used for iSCSI logins from this adapter.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

The **lsiscsi** command is executable only by root.

Examples

1. To list all iSCSI target data for the iSCSI TOE adapter `ics0`, enter:

   ```
   lsiscsi -l ics0
   ```

   The system displays output similar to the following:

   ```
   10.1.2.116 3260 iqn.sn9216.iscsi-hw1
   10.1.2.116 3260 iqn.sn2105.iscsi-target
   ```

2. To list all iSCSI target data for this host, enter:

   ```
   lsiscsi
   ```

   The system displays output similar to the following:

   ```
   ics0 1 10.1.2.116 3260 iqn.sn9216.iscsi-hw1
   ics0 10.1.2.116 3260 iqn.sn2105.iscsi-target
   ics1 11.23.45.67 iqn.mds9216.iscsi_hw2.116 3260 iqn.sn2105.iscsi-target
   ```

Location

/usr/sbin/lsiscsi

Files

Item	Description
src/bos/usr/sbin/iscsia	Contains the common source files from which the iSCSI commands are built.

Related information:

chiscsi command

rmiscsi command

---

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

Item	Description
-a	Specifies that all records in the **/etc/inittab** file are listed.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/etc/inittab	Indicates which processes the **init** command starts.

Related reference:
"mkitab Command" on page 728
Related information:
chitab command
init 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 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

Item	Description
/usr/bin/lskbd	Contains the **lskbd** command.
/usr/lib/nls/loc	Software keyboard map directory.

Related information:
chkbd command
smit command
Low Function Terminal (LFT) Subsystem Overview

lskst Command

Purpose

Lists the entries in the kernel security tables.

Syntax

lskst -t *table* [**-C** | **-f**] [*Name* [, *Name*]...]

lskst -l

Description

The **lskst** command reads the kernel security tables (KST) and displays the information on standard output (**stdout**). The output of the **lskst** command might differ from what is displayed by the **lsauth**, **lsrole** and **lssecattr** commands if the associated file databases are modified after the databases are sent to the KST through the **setkst** command.

Specify the table to be displayed with the **-t** flag. By default, all the information in the specified table is displayed. Alternatively, a specific entry in the table can be selected by specifying the *Name* parameter.

By default, the **lskst** command lists the attributes of each entry on one line. It displays attribute information as *Attribute* **=** *Value* definitions, each separated by a blank space. To list the table attributes in stanza format, use the **-f** flag. To list the information as colon-separated records, use the **-C** flag.

Flags

Item	Description

-C	Displays the table attributes in colon-separated records as follows:
	`#name:attribute1:attribute2:...`
	`entry_name:value1:value2:...`

-f	Displays the output in stanzas, with each stanza identified by the entry name. Each *Attribute* **=** *Value* pair is listed on a separate line: *entry_name*: *attribute1=value* *attribute2=value* *attribute3=value*
-l	Displays the current value of the `loglevel` variable that is set in the kernel by using the **setkst** command.
-t *table*	Retrieves data from the specified security table from the KST. The parameter for the **-t** flag can be one of the following values: **auth** Authorizations table **role** Role table **cmd** Privileged command table **dev** Privileged device table **dom** Domains **domobj** Domain objects

Parameters

Item	Description
Name	Represents a specific entry of a kernel table. It can be an authorization, a role, a privileged command or a privileged device, depending on the table specified by the **-t** *table* flag.

Security

The **lskst** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.kst.list	Required to run the command.

Examples

1. To retrieve all the entries in the role table from the KST, use the following command:
   ```
   lskst -t role
   ```
2. To display the entry for the **/usr/bin/mycmd** command from the privileged command table in stanza format, use the following command:
   ```
   lskst -t cmd -f /usr/bin/mycmd
   ```
3. To display the `aix.security` authorization table in the kernel, use the following command:
   ```
   lskst -t auth aix.security
   ```
4. To retrieve all the entries in the domain object table from the KST, use the following command:
   ```
   lskst -t domobj
   ```

Related information:

setkst command

/etc/security/authorizations file

/etc/security/privdevs file

RBAC

lsldap Command

Note: If an LDAP user is created with a UID > 2 ^ 31, the **lsldap** command displays it as a negative number.

Purpose

Displays naming service objects from the configured LDAP directory server.

Syntax

lsldap [**-a**] [*entity* [*entry_name* | *filter*]]

Description

The **lsldap** command displays the naming service objects from the configured LDAP directory server. It queries the LDAP server through the **secldapclntd** daemon. Some or all of the objects of a particular entity can be listed by the **lsldap** command. By default, the **lsldap** command displays only the distinguished name (DN) of the returned objects. In addition, the **-a** flag can be used to view the attributes.

The **lsldap** command supports the following entities:

Entity	objectClass	Default attribute name
aapolicies	ibm-aixAccountingAdminPolicy	ibm-aixAdminPolicyName
aaprojects	ibm-aixAccountingProject	ibm-aixProjectName
admkeystore	ibm-usrkeystore	cn
aixpert	ibm-aixAixpert	ibm-aixpertLabel
aliases	mailGroup	cn
auditclass	AIXAuditClassStanza	auditclassstanza
auditconfig	AIXAuditConfig	auditconfig

Entity	objectClass	Default attribute name
authorizations	ibm-authorization	cn
automount	automountMap nisObject	automountMapNamenisMapName
bootparams	bootableDevice	cn
domains	ibm-aixRBACdomain	ibm-aixRBACdomainName
domobjs	ibm-aixRBACdomainObject	ibm-aixRBACdomainObjectName
efscookies	ibm-efskscookies	cn
ethers	ieee802Device	cn
group	posixgroupAIXAccessGroup	cngroupname
grpkeystore	ibm-grpkeystore	cn
hosts	ipHost	cn
netgroup	ipNetgroup	cn
networks	ipNetwork	cn
passwd	posixAccountAIXAccount	uidusername
privcmds	ibm-privcmd	cn
privdevs	ibm-privdev	cn
protocols	ipProtocol	cn
roles	aixaccessroles	rolename
rpc	oncRpc	cn
services	ipService	cn
privfiles	ibm-privfile	cn
usrkeystore	ibm-usrkeystore	cn

The automount entity has two object classes. The **lsldap** command treats **automountMap** with higher precedence over **nisMap** by always returning **automountMap** objects if it finds any, and returning nisMap objects only in the absence of **automountMap** objects.

For the **passwd** and group entities, the **lsldap** command returns the correct objects according to the LDAP client configuration. However, the correct attribute name corresponding to the object classes must be supplied for `lsldap passwd attribute=value` queries.

If an entity name is not specified from the command line, the **lsldap** command displays container entries of the entities and any other entries that are siblings of these containers. Users must have root permissions to list the container entries.

The *entry_name* parameter is the name of the object to be queried. For example, if the entity is **passwd**, the *entry_name* is the user account name. The *entry_name* parameter is equivalent to `default attribute name = entry_name`. The **lsldap** command accepts the `*` wildcard in *entry_name* for a substring search. All entries are returned if *entry_name* is not specified.

Instead of *entry_name*, a *filter* can also be supplied to search for entries that match certain criteria. Simple filters can be specified as *attributename=attributevalue*, where *attributename* is the LDAP attribute name.

The **lsldap** command prints the result to **stdout**. If the **-a** flag is not specified, **lsldap** prints entries that are found in the form of DNs, with each DN separated by a blank line. If the **-a** flag is specified, each entry is printed in the ldif format, with a blank line between entries.

Flags

Item	Description
-a	Displays all attributes of returned objects. By default only the DN of the objects are displayed.

Exit Status

Upon success, the **lsldap** command returns `0`. Upon failure, a nonzero value is returned, with one of the following error messages that are written to **stderr**:

Item	Description
EIO	Connection error.
EINVAL	Invalid parameters.
EPERM	No permission to run the operation.
ENOMEM	Not enough memory.
other **errno**	Other errors.

Security

The **lsldap** command can be run by any user. It is owned by the root user and security group, and has access permissions of 555.

When a non-privileged user runs the `lsldap –a passwd` command for a `netgroup` enabled LDAP module, the `lsldap` command does not display the user information if the `DisplayNetgroupUserInfo` attribute is set to `no` in the `ldap.cfg` file. By default, the user information is displayed in a `netgroup` enabled LDAP module by running the `lsldap` command irrespective of your user privileges.

When you list the **passwd** entity with the **-a** flag by root user, **lsldap** returns all attributes of the found users. However, when the same command is run by a nonprivileged user, **lsldap** returns only the same commonly readable attributes as returned by the **lsuser** command in addition to the object class information. For all other entities, regardless of which user runs the command the same output is generated.

Examples

1. To list all entries of the host entity, enter the following command:

   ```
   lsldap hosts
   ```

   Information similar to the following is returned:

   ```
   dn: cn=myhost+ipHostNumber=192.3.193.46,ou=Hosts,cn=aixdata


   dn: cn=starfish+ipHostNumber=192.3.193.47,ou=Hosts,cn=aixdata


   dn: cn=loopback+ipHostNumber=127.0.0.1,ou=Hosts,cn=aixdata
   ```

2. To list host `starfish` and all of its attributes, enter the following command:

   ```
   lsldap -a hosts starfish
   ```

   Information similar to the following is returned:

   ```
   dn: cn=starfish+ipHostNumber=192.3.193.47,ou=Hosts,cn=aixdata

   objectClass: top

   objectClass: ipHost
   ```

```
objectClass: device

ipHostNumber: 192.3.193.47

cn: loopback

cn: localhost
```

3. To list users with names that begin with the letter b, enter the following command:
   ```
   lsldap passwd "b*"
   ```

   Information similar to the following is returned:
   ```
   dn: uid=bin,ou=people,cn=aixdata


   dn: uid=bob,ou=people,cn=aixdata
   ```
4. To list user foo and its attributes, enter the following command:
   ```
   lsldap -a passwd foo
   ```

   Information similar to the following is returned:
   ```
   dn: uid=foo,ou=people,cn=aixdata

   uid: foo

   objectClass: account

   objectClass: posixAccount

   objectClass: shadowAccount

   objectClass: top

   cn: foo

   uidNumber: 259

   gidNumber: 1

   homeDirectory: /home/foo

   loginShell: /usr/bin/ksh

   shadowlastchange: 12740

   userpassword: {crypt}rNnLQ9TAD2u/k

   shadowmin: 5
   ```
5. To list users who run **/usr/bin/ksh**, enter the following command:
   ```
   lsldap passwd loginshell=/usr/bin/ksh
   ```

   Information similar to the following is returned:
   ```
   dn: uid=bin,ou=people,cn=aixdata


   dn: uid=bob,ou=people,cn=aixdata


   dn: uid=foo,ou=people,cn=aixdata
   ```

Restrictions

The **lsldap** command relies on the **secldapclntd** daemon to work.

Location

/usr/sbin/lsldap

Related information:

secldapclntd command

Light Directory Access Protocol

---

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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
```

Output 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 reference:

"monitord Daemon" on page 877

Related information:

chlicense command

lslpclacl Command

Purpose

Displays the access controls for the least-privilege (LP) resource class (**IBM.LPCommands**).

Syntax

To display the access controls for the **IBM.LPCommands** resource class:

- On the local node:
  lslpclacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On all nodes in a domain:
  lslpclacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On a subset of nodes in a domain:
  lslpclacl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

Description

The **lslpclacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) resource class (**IBM.LPCommands**). The accesses contained in the ACL entries are displayed. The **IBM.LPCommands** Class ACL controls access to the **IBM.LPCommands** class operations. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description
Identity	The network identity of the user. See the lpacl command for a description of the network identity.

Field	Description	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays the **IBM.LPCommands** Class ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpclacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpclacl –a** runs in the management domain. To run **lslpclacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpclacl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). It is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). 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 pipe symbol (|) when the information that you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[**,***host2*,**...**]
Specifies the node in the domain from which the **IBM.LPCommands** Class ACL is displayed. By default, the **IBM.LPCommands** Class ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** 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 runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where 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, unless the **-a** flag or the **-n** flag is specified.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpclacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 274 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpclacl

Examples

1. To list the **IBM.LPCommands** Class ACLs on **nodeA** in table format, run this command on **nodeA**:
   ```
   lslpclacl
   ```
   The following output is displayed:
   ```
   Identity                        Permissions     NodeName

   joe@LOCALHOST                   ra              nodeA

   bill@0x374bdcbe384ed38a         rwa             nodeA

   jane@0x374bdcbe384ed38a         rwa             nodeA
   ```
2. To list the **IBM.LPCommands** Class ACLs on **nodeA** in long format, run this command on **nodeA**:
   ```
   lslpclacl -l
   ```
   The following output is displayed:
   ```
   Class ACLs for LPRM

   NodeName nodeA

       Identity    =     joe@LOCALHOST

       Permissions =     ra
   ```

```
Identity    =     bill@0x374bdcbe384ed38a

Permissions =     rwa


Identity    =     jane@0x374bdcbe384ed38a

Permissions =     rwa
```

3. To list the **IBM.LPCommands** Class ACLs on **nodeA** in delimited format, run this command on **nodeA**:

```
lslpclacl -d
```

The following output is displayed:

```
Identity|Permissions|NodeName

joe@LOCALHOST|ra|nodeA

bill@0x374bdcbe384ed38a|rwa|nodeA

jane@0x374bdcbe384ed38a|rwa|nodeA
```

4. To list the **IBM.LPCommands** Class ACLs on **nodeA** in the active domain, run this command:

```
lslpclacl -a
```

The following output is displayed:

```
Identity                        Permissions     NodeName

joe@LOCALHOST                   ra              node1.pok.ibm.com

bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com

jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com

joe@LOCALHOST                   ra              node2.pok.ibm.com

jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com
```

5. To list the **IBM.LPCommands** Class ACLs on **nodeA** in the active domain and list the peer domain name, run this command:

```
lslpclacl -ap
```

The following output is displayed:

```
Identity                        Permissions     NodeName                PeerDomain

joe@LOCALHOST                   ra              node1.pok.ibm.com       PD1

bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1

joe@LOCALHOST                   ra              node2.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com       PD1
```

Related reference:

"lsaudrec Command" on page 359

"lscondition Command" on page 380

"lscondresp Command" on page 385

Related information:

rmccli command

lslpcmd Command

Purpose

Lists information about the least-privilege (LP) resources on one or more nodes in a domain.

Syntax

To display LP resource information:

- On the local node:

  lslpcmd [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] [**-h**] [**-TV**]

- On all nodes in a domain:

  lslpcmd -a [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] [**-h**] [**-TV**]

- On a subset of nodes in a domain:

  lslpcmd -n *host1* [**,***host2***,**...] [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] [**-h**] [**-TV**]

Description

The **lslpcmd** command displays information about LP resources on one or more nodes in a domain. LP resources are **root** commands or scripts to which users are granted access based on permissions in the LP access control lists (ACLs). Use this command to display the attributes of one or more LP commands by specifying the *resource_name1*,[*resource_name2*,...] parameter. If you omit this parameter, the **lslpcmd** command lists the names of all of the LP commands. Use the **–A** flag to list all of the LP commands and all of their attributes and values. Use the **–R** flag to list one or more LP resources that have a particular **RunCmdName** value.

The **lslpcmd** command lists the following information about defined LP resources:

Field	Description
Name	The name of the LP resource.
CommandPath	The fully-qualified path of the LP resource.
Description	A description of the LP resource.
Lock	The lock setting. Valid values are: **0** (the lock is not set) and **1** (the lock is set).
CheckSum	The **CheckSum** value of the LP resource to which **CommandPath** points. The LP resource manager assigns a value of **0** if the LP resource does not exist or if the user did not update the **CheckSum** value after the LP resource was made available.
RunCmdName	The LP resource name that is used as a parameter with the **runlpcmd** command.
FilterScript	The path to the filter script.
FilterArg	The list of arguments to pass to **FilterScript**.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays information about one or more LP resources on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable's setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpcmd** command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpcmd –a** runs in the management domain. To run **lslpcmd –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-n *host1***[,***host2***,...]**
Specifies the node or nodes in the domain on which the LP resource is to be listed. By default, the LP resource is changed on the local node. The **–n** flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:
1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpcmd** command runs once for the first valid scope that the LP resource manager finds.

–A Displays all of the LP resources with their attributes and values.

–R Display all attributes of the LP resources that have the same **RunCmdName** value.

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

-T Writes the command's trace messages to standard error.

-V Writes the command's verbose messages to standard output.

Parameters

*resource_name1***[,***resource_name2***,...]**
Specifies one or more LP resources for which you want to display information.

Security

To run the **lslpcmd** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT

Determines the system that is used for the session with the RMC daemon. 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 LP resources that are processed.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-V** flag is specified, this command's 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 LP resources on the local node, enter:

   ```
   lslpcmd
   ```

   The output will look like this:

   ```
   lpcommand1
   lpcommand2
   ```

2. To list the names and attributes of all LP resources on the local node, enter:

   ```
   lslpcmd -A
   ```

   The output will look like this:

   ```
   Name=lpcommand1
   CommandPath=/tmp/my_command
   Description=
   Lock=1
   CheckSum=112
   RunCmdName=lpcommand1
   FilterScript=
   FilterArg=
   ---------------------------------
   Name=lpcommand2
   CommandPath=/tmp/cmds/this_command
   Description=
   ```

```
Lock=0
CheckSum=0
RunCmdName=lpcommand2
FilterScript=
FilterArg=
---------------------------------
```

3. To list the attributes of the LP resource **lpcommand1** on the local node, enter:

```
lslpcmd lpcommand1
```

   The output will look like this:

```
Name=lpcommand1
CommandPath=/tmp/my_command
Description=
Lock=1
CheckSum=100
RunCmdName=lpcommand1
FilterScript=
FilterArg=
```

4. To list the attributes of LP resources that have a **RunCmdName** value of **rpower** on the local node, enter:

```
lslpcmd -R rpower
```

   The output will look like this:

```
Name=lpcommand1
CommandPath=/opt/csm/bin/rpower
Description=
Lock=1
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
FilterArg=node1,node2,node3
------------------------------
Name=lpcommand2
CommandPath=/opt/csm/bin/rpower
Description=
Lock=0
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
FilterArg=node4,node5,node6
------------------------------
⋮
```

Location

/opt/rsct/bin/lslpcmd
Contains the **lslpcmd** command

---

lslpp Command

Purpose

Lists installed software products.

Syntax

lslpp [**-R** { *path* | **ALL** }] { **-d** | **-E** | **-f** | **-h** | **-i** | **-l** | **-L** | **-p** } [**-a**] [**-c**] [**-J**] [**-q**] [**-I**] [**-O** { [**r**] [**s**] [**u**] }] [*FilesetName ...* | **-b** *File* | **all**]

lslpp [**-R** { *path* | **ALL** }] **-w** [**-c**] [**-q**] [**-O** { [**r**] [**s**] [**u**] }] [*FileName ...* | **all**]

lslpp [**-R** { *path* | **ALL** }] **-L -c** [**-v**]

lslpp [**-R** { *path* | **ALL** }] **-S** [**A**|**O**]

lslpp [**-R** { *path* | **ALL** }] **-e**

Description

The **lslpp** command displays information about installed filesets or fileset updates. The *FilesetName* parameter is the name of a software product. The *File* parameter specifies a bundle file to use as a fileset list.

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 **-I** (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 **lslpp** command and the **compare_report** command both show information about interim fixes installed on the system. The **lslpp -L** or **lslpp -Lc** command and the **lslpp -e** command must be run by root. Any interim fix information returned is used by the **compare_report** command. The information includes an interim fix label and a level value. The interim fix label is the equivalent of a fileset name, and its level is based on the time (*YY.MM.DD.HHMMSS*, where *YY* is the year, *MM* is the month, *DD* is the day, *HH* is the hour, *MM* is the minute, and *SS* is the second) in which the interim fix was packaged. If a non-root user runs these commands, only software products and levels are returned, and interim fix information is not included. If a root user runs the **lslpp -e** command and the **lslpp -L** command, interim fix information can be shown.

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 *FilesetName* parameter. 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.
EFIXLOCKED	The specified fileset is installed on the system and is locked by the interim fix manager (the **emgr** command).
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

Item	Description
-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.
-b *File*	Specifies a bundle file to search for fileset names. The filesets listed in the bundle are then listed as if they had been specified explicitly as *FilesetName* parameters. To mimic installp behavior, the installp image names are automatically wildcarded. For example, a bundle file entry of `I:bos.abc` will behave as if `bos.abc*` was specified as a *FilesetName* parameter. **Note:** This might also return results for `bos.abcdef`.If the file does not reside in one of the known bundle locations, the full path and file name, including extension, must be specified.
-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 interim fix installed on the system.
-E	Lists license agreements.

Item	Description
-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, type, and a 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 or technology level for the specified filesets. In addition, this flag lists any subsystem selective fixes that were installed on top of the maintenance or technology level. RPM and ISMP images are also listed. When combined with the **-c** flag, there is a difference in the **Type** field when used with an installp image. A blank value indicates an installp image without any updates. A value of **F** indicates an installp image with updates. When combined with the **-c** flag, the build date, which is specified by the year and the week in the form of **yyww** (for example, 0852), is displayed for the fileset, if there is one. Additional fields are displayed with the **-Lc** output, as indicated in the header of the output.
-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.
-R { *path* | **ALL** }	Indicates a user-specified installation location.
-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 only information from the vendor database, which contains ISMP product information. This flag is only valid when used with both the **-L** and the **-c** flags.
-w	Lists fileset that owns this file.

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
   ```

9. To display the installation state for all the filesets contained in the Server bundle located at **/usr/sys/inst.data/sys_bundles/Server.bnd**, type:

   ```
   lslpp -L -b Server
   ```

   or:

   ```
   lslpp -L -b /usr/sys/inst.data/sys_bundles/Server.bnd
   ```

Files

Item	Description
/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.
/usr/sys/inst.data/sys_bundles/ and /usr/sys/inst.data/user_bundles/	Known locations for bundle files. Bundle files should have a **.bnd** extension.

Related reference:

"installp Command" on page 62

"inulag Command" on page 78

Related information:
emgr command
nim command

lslpracl Command

Purpose

Displays the access controls for a least-privilege (LP) resource.

Syntax

To display the access controls for an LP resource:

- On the local node:
  lslpracl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-L]** **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]** [*name*]
- On all nodes in a domain:
  lslpracl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-L]** **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]** [*name*]
- On a subset of nodes in a domain:
  lslpracl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-L]** **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]** [*name*]

Description

The **lslpracl** command displays the access control list (ACL) that is associated with a least-privilege (LP) resource. The accesses contained in the ACL entries are displayed. The Resource ACL controls access to the LP resources. If no LP resource name is specified, the Resource ACLs for all LP resources are listed. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description	
Name	The name of the LP resource. See "lpacl Information" on page 274 for a description of the network identity.	
Identity	The network identity of the user.	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the LP resource (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the LP resource is defined. This field is displayed when the **-p** flag is specified.	

If the Resource ACL indicates that the Resource Shared ACL controls access to the LP resource, the ID is displayed as **Uses Resource Shared ACL** and there is no permission value. Use the **-L** flag to display the Resource Shared ACL when it is used by the Resource ACLs that are being displayed.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Parameters

name Specifies the name of the LP resource.

Flags

-a Displays the Resource ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:
1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpracl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpracl –a** runs in the management domain. To run **lslpracl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpracl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). It is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). 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 pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1***[,***host2***,...]**
Specifies the node in the domain from which the Resource ACL is displayed. By default, the Resource ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** 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 runs once for the first valid scope found.

-L Displays the accesses of the Resource Shared ACL if the Resource ACL indicates that access is controlled by the Resource Shared ACL.

-p Displays the name of the RSCT peer domain in which the LP resource is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where 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, unless the **-a** flag or the **-n** flag is specified.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpracl** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 274 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpracl

Examples

1. To list the Resource ACLs for the LP resource **lpcommand1** on **nodeA** in table format, run this command on **nodeA**:

   ```
   lslpracl lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM

   Name            Identity                Permissions     NodeName

   lpcommand1      joe@LOCALHOST           rx              nodeA

   lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA

   lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA
   ```

2. To list the Resource ACLs for the LP resource **lpcommand1** on **nodeA** in long format, run this command on **nodeA**:

   ```
   lslpracl -l lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM

   Name lpcommand1, NodeName nodeA

       Identity    =     joe@LOCALHOST

       Permissions =     rx


       Identity    =     bill@0x374bdcbe384ed38a

       Permissions =     rx
   ```

```
Identity    =     jane@0x374bdcbe384ed38a

Permissions =     rwax
```

3. To list the Resource ACLs for the LP resource **lpcommand1** on **nodeA** in delimited format, run this command on **nodeA**:

   ```
   lslpracl -d lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM

   Name|Identity|Permissions|NodeName

   lpcommand1|joe@LOCALHOST|rx|nodeA

   lpcommand1|bill@0x374bdcbe384ed38a|rx|nodeA

   lpcommand1|jane@0x374bdcbe384ed38a|rwax|nodeA
   ```

4. To list the Resource ACLs for the LP resource **lpcommand1** in the active domain, run this command on **nodeA**:

   ```
   lslpracl -a lpcommand1
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM

   Name            Identity                Permissions     NodeName

   lpcommand1      joe@LOCALHOST           rx              nodeA.pok.ibm.com

   lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA.pok.ibm.com

   lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA.pok.ibm.com

   lpcommand1      joe@LOCALHOST           rx              nodeB.pok.ibm.com

   lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeB.pok.ibm.com
   ```

5. To list the Resource ACLs for all LP resources on **nodeA**, run this command on **nodeA**:

   ```
   lslpracl
   ```

   The following output is displayed:

   ```
   Resource ACLs for LPRM

   Name            Identity                Permissions     NodeName

   lpcommand1      joe@LOCALHOST           rx              nodeA

   lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA

   lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA

   lpcommand2      jim@LOCALHOST           rx              nodeA

   lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA

   lpcommand3      mary                    rwax            nodeA

   lpcommand4      bob@LOCALHOST           rx              nodeA

   lpcommand4      sam@0x374bdcbe384ed38a  rwax            nodeA
   ```

6. To list the Resource ACLs for the LP resource **lpcommand1** in the active domain and list the peer domain name, run this command on **nodeA**:

```
lslpracl -ap lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permission NodeName             PeerDomain

lpcommand1      joe@LOCALHOST           rx         nodeA.pok.ibm.com    PD1

lpcommand1      bill@0x374bdcbe384ed38a rx         nodeA.pok.ibm.com    PD1

lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeA.pok.ibm.com    PD1

lpcommand1      joe@LOCALHOST           rx         nodeB.pok.ibm.com    PD1

lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeB.pok.ibm.com    PD1
```

7. To list the Resource ACLs for the LP resource **lpcommand2** on **nodeA**, run this command on **nodeA**:

```
lslpracl lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                   Permissions   NodeName

lpcommand2      Uses Resource Shared ACL                 nodeA
```

8. To list the Resource ACLs for the LP resource **lpcommand2** on **nodeA**, and show the Resource Shared ACL if it is used, run this command on **nodeA**:

```
lslpracl -L lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permissions     NodeName

lpcommand2      bill@0x374bdcbe384ed38a rx              nodeA

lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA
```

Related reference:

"lslpcmd Command" on page 429

Related information:

chlpclacl command

rmlpcmd command

runlpcmd command

lslpriacl Command

Purpose

Displays the access controls for the least-privilege (LP) Resource Initial ACL.

Syntax

To display the access controls for the Resource Initial ACL:

- On the local node:
  lslpriacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On all nodes in a domain:

lslpriacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

- On a subset of nodes in a domain:

  lslpriacl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

Description

The **lslpriacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Initial ACL. The accesses contained in the ACL entries are displayed. The Resource Initial ACL is used as the Initial ACL that gets copied to the Resource ACL when an LP resource is created. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description	
Identity	The network identity of the user. See "lpacl Information" on page 274 for a description of the network identity.	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays the Resource Initial ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpriacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the

CT_MANAGEMENT_SCOPE environment variable is not set. In this case, **lslpriacl –a** runs in the management domain. To run **lslpriacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpriacl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). This is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). 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 pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1***[,***host2***,...]**
Specifies the node in the domain from which the Resource Initial ACL is displayed. By default, the Resource Initial ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** 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 runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where 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, unless the **-a** flag or the **-n** flag is specified.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpriacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 274 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslpriacl

Examples

1. To list the Resource Initial ACLs on **nodeA** in table format, run this command on **nodeA**:

   ```
   lslpriacl
   ```

   The following output is displayed:

   ```
   Resource Initial ACLs for LPRM

   Identity                       Permissions     NodeName

   joe@LOCALHOST                  rx              nodeA
   ```

```
bill@0x374bdcbe384ed38a         rwx             nodeA

jane@0x374bdcbe384ed38a         rwax            nodeA
```

2. To list the Resource Initial ACLs on **nodeA** in long format, run this command on **nodeA**:
```
lslpriacl -l
```

   The following output is displayed:
```
Resource Initial ACLs for LPRM

NodeName c175n06.ppd.pok.ibm.com

   Identity    =     joe@LOCALHOST

   Permissions =     rx


   Identity    =     bill@0x374bdcbe384ed38a

   Permission  =     rwx


   Identity    =     jane@0x374bdcbe384ed38a

   Permissions =     rwax
```

3. To list the Resource Initial ACLs on **nodeA** in delimited format, run this command on **nodeA**:
```
lslpriacl -d
```

   The following output is displayed:
```
Resource Initial ACLs for LPRM

Identity|Permissions|NodeName

joe@LOCALHOST|rx|nodeA

bill@0x374bdcbe384ed38a|rwx|nodeA

jane@0x374bdcbe384ed38a|rwax|nodeA
```

4. To list the Resource Initial ACLs in the active domain, run this command:
```
lslpriacl -a
```

   The following output is displayed:
```
Resource Initial ACLs for LPRM

Identity                        Permissions     NodeName

joe@LOCALHOST                   rx              nodeA.pok.ibm.com

bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com

jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com

joe@LOCALHOST                   rx              nodeB.pok.ibm.com

jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com
```

5. To list the Resource Initial ACLs in the active domain and list the peer domain name, run this command:
```
lslpriacl -ap
```

   The following output is displayed:

```
Resource Initial ACLs for LPRM

Identity                        Permissions     NodeName                PeerDomain

joe@LOCALHOST                   rx              nodeA.pok.ibm.com       PD1

bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com       PD1

joe@LOCALHOST                   rx              nodeB.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com       PD1
```

Related reference:

"lslpcmd Command" on page 429

"lslpracl Command" on page 437

Related information:

runlpcmd command

---

lslprsacl Command

Purpose

Displays the access controls for the least-privilege (LP) Resource Shared ACL.

Syntax

To display the access controls for the Resource Shared ACL:

- On the local node:
  lslprsacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On all nodes in a domain:
  lslprsacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On a subset of nodes in a domain:
  lslprsacl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

Description

The **lslprsacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Shared ACL. The accesses contained in the ACL entries are displayed. The Resource Shared ACL controls access to LP resources in which the Resource ACL indicates that the Resource Shared ACL is used. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description
Identity	The network identity of the user. See "lpacl Information" on page 274for a description of the network identity.

Field	Description	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays Resource Shared ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslprsacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslprsacl –a** runs in the management domain. To run **lslprsacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlprsacl** command.

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). 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 pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[**,***host2*,**...**]
Specifies the node in the domain from which the Resource Shared ACL is displayed. By default, the Resource Shared ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** 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 runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where 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, unless the **-a** flag or the **-n** flag is specified.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslprsacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 274 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for the AIX and Linux operating systems.

Location

/opt/rsct/bin/lslprsacl

Examples

1. To list the Resource Shared ACLs on **nodeA** in table format, run this command on **nodeA**:
   ```
   lslprsacl
   ```
   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   Identity                        Permissions     NodeName

   joe@LOCALHOST                   rx              nodeA

   bill@0x374bdcbe384ed38a         rwx             nodeA

   jane@0x374bdcbe384ed38a         rwax            nodeA
   ```
2. To list the Resource Shared ACLs on **nodeA** in long format, run this command on **nodeA**:
   ```
   lslprsacl -l
   ```
   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   NodeName c175n06.ppd.pok.ibm.com

      Identity    =     joe@LOCALHOST

      Permissions =     rx
   ```

```
Identity    =     bill@0x374bdcbe384ed38a

Permissions =     rwx


Identity    =     jane@0x374bdcbe384ed38a

Permissions =     rwax
```

3. To list the Resource Shared ACLs on **nodeA** in delimited format, run this command on **nodeA**:
   ```
   lslprsacl -d
   ```

   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   Identity|Permissions|NodeName

   joe@LOCALHOST|rx|nodeA

   bill@0x374bdcbe384ed38a|rwx|nodeA

   jane@0x374bdcbe384ed38a|rwax|nodeA
   ```
4. To list the Resource Shared ACLs in the active domain, run this command:
   ```
   lslprsacl -a
   ```

   The following output is displayed:
   ```
   Identity                      Permissions     NodeName

   joe@LOCALHOST                 rx              nodeA.pok.ibm.com

   bill@0x374bdcbe384ed38a       rwx             nodeA.pok.ibm.com

   jane@0x374bdcbe384ed38a       rwax            nodeA.pok.ibm.com

   joe@LOCALHOST                 rx              nodeB.pok.ibm.com

   jane@0x374bdcbe384ed38a       rwax            nodeB.pok.ibm.com
   ```
5. To list the Resource Shared ACLs in the active domain and list the peer domain name, run this command:
   ```
   lslprsacl -ap
   ```

   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   Identity                      Permissions     NodeName              PeerDomain

   joe@LOCALHOST                 rx              nodeA.pok.ibm.com     PD1

   bill@0x374bdcbe384ed38a       rwx             nodeA.pok.ibm.com     PD1

   jane@0x374bdcbe384ed38a       rwax            nodeA.pok.ibm.com     PD1

   joe@LOCALHOST                 rx              nodeB.pok.ibm.com     PD1

   jane@0x374bdcbe384ed38a       rwax            nodeB.pok.ibm.com     PD1
   ```

Related reference:

"lslpriacl Command" on page 442

"mklpcmd Command" on page 736
Related information:
chlprsacl command
runlpcmd 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 System Management Interface Tool (SMIT) **smit lslv** fast path to run this command.

Flags

Item	Description
-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.

Item	Description
-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:

Item	Description
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 (LV identifier)	Identifier of the logical volume.
Permission	Access permission; read-only or read-write.
Volume group state (VG 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 (LV 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 on or off.
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 (Sched 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.
BB policy	Bad block relocation policy.
Inter-policy	Inter-physical allocation policy.
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.
Each LP copy on a separate PV?	The strictness value. 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, it 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.
Serialize IO?	Serialization of overlapping IOs state of yes or no. If serialization is turned on (yes), then overlapping IOs are not allowed on a block range, and only a single IO in a block range is processed at any one time. Most applications, such as file systems and databases, perform serialization; therefore, serialization should be turned off (no). The default setting for new logical volumes is no.
Stripe width	The number of physical volumes being striped across.
Stripe size	The number of bytes per stripe.
Infinite Retry	Lists the infinite retry option of the logical volume.
Preferred Read	Lists the preferred logical volume copy for the read operation.

Examples

1. To display information about the testlv logical volume, enter:

   ```
   lslv testlv
   ```

   Information about the testlv logical volume, its logical and physical partitions, and the volume group to which it belongs is displayed as shown in the following example output:

```
LOGICAL VOLUME:     testlv                  VOLUME GROUP:   testvg
LV IDENTIFIER:      000e8b6e0000d900000001476c303bc8.1 PERMISSION:     read/write
VG STATE:           active/complete         LV STATE:       closed/syncd
TYPE:               jfs                     WRITE VERIFY:   off
MAX LPs:            512                     PP SIZE:        128 megabyte(s)
COPIES:             3                       SCHED POLICY:   parallel
LPs:                10                      PPs:            30
STALE PPs:          0                       BB POLICY:      relocatable
INTER-POLICY:       minimum                 RELOCATABLE:    yes
INTRA-POLICY:       middle                  UPPER BOUND:    32
MOUNT POINT:        N/A                     LABEL:          None
MIRROR WRITE CONSISTENCY: on/ACTIVE
EACH LP COPY ON A SEPARATE PV ?: yes
Serialize IO ?:     NO
INFINITE RETRY:     no                      PREFERRED READ: 3
```

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

Item	Description
/usr/sbin	Contains the **lslv** command.

Related reference:

"lsvg Command" on page 554

Related information:

varyonvg command

Monitoring disk I/O

Logical volume storage

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 System Management Interface Tool (SMIT) **smit lsmaster** fast path to run this command.

Flags

Item	Description
-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

Item	Description
/var/yp/*domainname* directory	Contains the NIS maps for the NIS domain.

Related information:

chmaster command

smit command

Network File System (NFS) Overview for System Management

Reference command

---

lsmcode Command

Purpose

Displays microcode and firmware levels of the system and adapters and devices.

Syntax

lsmcode [-A | -d *Name* **] [-r | -c] [-t [service | system | adapter | adapter-boot | raid-dasd | backplane]]**

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

Item	Description
-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
-t	Specifies the microcode type. The microcode level information of the specified type is displayed. You can use the **-t** flag only when a device supports multiple types. You can use the **-A** flag to find any devices supporting multiple types.

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
   .....
   ```

6. To display the microcode level of the **adapter** microcode type for a RAID adapter **scraid0**, type:

```
lsmcode -rd scraid0 -t adapter
```

The system displays a message similar to the following:

```
adapter:4.50.01
```

Files

Item	Description
/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**] [**-n**] [**-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, USB disks, or tapes.

The **lsmksysb -r** and **restorevgfiles** commands perform identical operations and must be considered interchangeable.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You might 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 defaults to 100.
-B	Prints the volume group backup log to **stdout**. This flag displays 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 that is used to make backup, date, shrink size, full size of the backup, and recommended maintenance or technology level (if any). **Note:** The shrink size is the size of the data on all file systems. The full size is total size of each file system (unused + data).
-c	Produces colon-separated output. This flag works only with the **-l** and **-L** flags.
-d *path*	Specifies the directory path to which the files are 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.
-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* defaults to **/dev/rmt0**.

Item	Description
-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 or technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused + data). The **-l** flag also displays the logical volume and file system 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**.
-n	Does not restore ACLs, PCLs, or extended attributes
-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 are restored. If the **-r** flag is not used, then executing the **lsmksysb** command lists only 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

Item	Description
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 must be specified in the space-separated list. All files in the specified directory are 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 that is on the default device **/dev/rmt0**, enter the following command:

   ```
   lsmksysb
   ```

2. To list the contents of the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lsmksysb -f /dev/cd1
   ```

3. To list the contents of the system backup that is on device **/dev/cd1**, which is a user volume group that is not rootvg, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -s
   ```

4. To restore **/etc/filesystems** from the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -r ./etc/filesystems
   ```

5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter the following command:

   ```
   lsmksysb -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```

6. To display colon separated lpp information about a **mksysb** backup tape that is on **/dev/rmt0**, enter the following command:

   ```
   lsmksysb -Lc -f /dev/rmt0
   ```

7. To display the backup log of the volume group to **stdout**, enter the following command:

   ```
   lsmksysb -B
   ```

8. To list volume group and general backup data about a backup that is on **/tmp/mybackup**, enter the following command:

   ```
   lsmksysb -l -f /tmp/mybackup
   ```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter the following command:

   ```
   lsmksysb -V -f /dev/rmt0
   ```

10. To list the contents of the system backup that is on device **/dev/usbms0,** enter the following command:

    ```
    lsmksysb –f /dev/usbms0
    ```

Files

Item	Description
/usr/bin/lsmksysb	Contains the **lsmksysb** command

Related information:

restorevgfiles command

---

lsmp Command

Purpose

Displays mirror pool information.

Syntax

lsmp [**-L**] [**-A**] [**-n** *descriptorphysicalvolume*] [**-m** *mirrorpoolname*] *vgname*

Description

The **lsmp** command displays mirror pool information for a volume group.

Note: The volume group must be varied on to list the mirror pool information. If the volume group is varied off, you must specify the source disk name using the **-n** flag to list the information.

Flags

Item	Description
-A	Displays information specific to asynchronous mirroring, if it exists.
-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 data.
-m *mirrorpoolname*	Displays mirror pool information only for the mirror pool that is specified by the *mirrorpoolname* parameter. If you do not specify the flag, information for all mirror pools that belong to the volume group is displayed.
-n *descriptorphysicalvolume*	Accesses information from the descriptor area that is specified by the *descriptorphysicalvolume* variable. The information might not be current because 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 current information is displayed. The volume group does not need to be active when you use the **-n** flag.

Parameters

Item	Description
vgname	Specifies the volume group name.

Examples

1. To display all mirror pool information for a volume group, enter the following command:

   ```
   lsmp vg1
   ```

   The following output is displayed:

   ```
   VOLUME GROUP:      vg1                 Mirror Pool Super Strict: no

   MIRROR POOL:       mp1                 Mirroring Mode:    SYNC
   MIRROR POOL:       mp2                 Mirroring Mode:    SYNC
   ```

2. To display all mirror pool information for a volume group and include asynchronous mirroring information in the output, enter the following command:

   ```
   lsmp -A glvm_vg
   ```

   The following output is displayed:

   ```
   VOLUME GROUP:      glvm_vg             Mirror Pool Super Strict: yes

   MIRROR POOL:       mp_bvr              Mirroring Mode:    ASYNC
   ASYNC MIRROR STATE:  inactive          ASYNC CACHE LV:      mp_pok_lv
   ASYNC CACHE VALID:   yes               ASYNC CACHE EMPTY:   yes
   ASYNC CACHE HWM:     60                ASYNC DATA DIVERGED: no

   MIRROR POOL:       mp_pok              Mirroring Mode:    ASYNC
   ASYNC MIRROR STATE:  active            ASYNC CACHE LV:      bvr_pok_lv
   ASYNC CACHE VALID:   yes               ASYNC CACHE EMPTY:   no
   ASYNC CACHE HWM:     90                ASYNC DATA DIVERGED: no
   ```

3. To display information for the mirror pool `mp_pok` from the `glvm_vg` volume group and include asynchronous mirroring information in the output, enter the following command:

   ```
   lsmp -A –m mp_pok glvm_vg
   ```

   The following output is displayed:

   ```
   VOLUME GROUP:      glvm_vg             Mirror Pool Super Strict: yes

   MIRROR POOL:       mp_pok              Mirroring Mode:    ASYNC
   ASYNC MIRROR STATE:  active            ASYNC CACHE LV:      bvr_pok_lv
   ASYNC CACHE VALID:   yes               ASYNC CACHE EMPTY:   no
   ASYNC CACHE HWM:     90                ASYNC DATA DIVERGED: no
   ```

Related information:

chmp command

PowerHA SystemMirror for Geographic LVM

Mirror Pools

---

lsmpio Command

Purpose

Display information about the MultiPath I/O (MPIO) storage devices.

Syntax

lsmpio [**-l** *device_name*]

lsmpio -S [**-l** *device_name*] [**-d**]

lsmpio -z [**-l** *device_name*]

lsmpio -q [**-l** *device_name*]

lsmpio -a [**-r**] [**-e**] [**-z**]

lsmpio -h

Description

The **lsmpio** command displays information that is related to AIX MPIO storage devices. This command works only for devices that are controlled by path-control modules (PCMs) that are enabled for **lsmpio** support.

There are four types of information that the **lsmpio** command displays:

- Path operational status
- Path statistics
- Device inquiry data
- Parent adapter information

The **lsmpio** command can also be used to reset the path statistics to zero.

Displaying path operational status

If the **lsmpio** command is run without any flags or with the **-l** flag, it displays the path operational status. This output is similar the output displayed by running the **lspath** command, with an additional **Extended Status** field:

```
lspath -F "path_id status parent connection"
```

If the **-l** flag is not included, the status is shown for all MPIO paths in the system. If the **-l** flag and device name are included, the status is shown only for those paths that are used to attach the specified device.

The **-e** flag displays error count information for the local adapter ports and remote ports. These error counts indicate an issue with the physical connectivity between the local adapter and the remote storage port. The error counts indicate recent errors for different time ranges.

The **Path Status** column shows the same status as the **lspath** command. The valid values of status are *Enabled*, *Disabled*, *Failed*, or *Missing*.

The **Extended Status** field might contain one or more three-letter status abbreviations to provide more detailed path status.

Note: Not all extended status types are applicable to all devices.

The possible values for the **Extended Status** field follow:

Opt Indicates that the path is an optimized path. This value indicates a path that attaches to a preferred controller in a device that has multiple controllers. The PCM selects one of the preferred paths for I/O operations, whenever possible.

Non Indicates that the path is a non-optimized path. On a device with preferred paths, this path is not considered as preferred path. The PCM avoids the selection of this path for I/O operations, unless all preferred paths fail.

Act Indicates that the path is an active path on a device that has active and passive controllers. The PCM selects active paths for I/O operations on such a device.

Pas Indicates that the path is a passive path on a device that has active and passive controllers. The PCM avoids the selection of passive paths.

Sel Indicates that the path is being selected for I/O operations, for the time when the **lsmpio** command is to be run.

Rsv Indicates that the path has experienced an unexpected reservation conflict. This value might indicate a usage or configuration error, with multiple hosts accessing the same disk.

Fai Indicates that the path experienced a failure. It is possible for a path to have a **Path Status** value of *Enabled* and still have an **Extended Status** value of *Fai*. This scenario indicates that operations sent on this path are failing, but AIX MPIO has not marked the path as *Failed*. In some cases, AIX MPIO leaves one path to the device in *Enabled* state, even when all paths are experiencing errors.

Deg Indicates that the path is in a degraded state. This scenario indicates that the path was being used for I/O operations. Those operations experienced errors, thus causing the PCM to temporarily avoid the use of the path. Any additional errors might cause the path to fail.

Clo Indicates that the path is closed. If all paths to a device are closed, the device is considered to be closed. If only some paths are closed, then those paths might have experienced errors during the last time the device was opened. The AIX MPIO periodically attempts to recover closed paths, until the device path is open.

Displaying path statistics

The **-S** flag, along with the optional **-d** flag, causes the **lsmpio** command to display normal or detailed path statistics. The optional **-l** flag allows the user to restrict the display to contain statistics for just one MPIO storage device. The statistics include how many times the path has been selected for an I/O operation, how many errors have occurred on the path, and how many times the path has failed. The detailed statistics information breaks down the failure counts, into counts of different types of failures.

Displaying device inquiry data

The **-q** flag of the **lsmpio** command causes the AIX MPIO to query the device, by using Small Computer System Interface (SCSI) commands to retrieve and display information about the attached device. Because each queried device is opened and queried by using SCSI commands, this operation might take time to run when there are many devices.

Displaying parent adapter information

The **-a** flag, along with the optional **-r** flag, causes the **lsmpio** command to display information about the Fibre Channel adapters that are used by the AIX MPIO storage devices. The information includes the details about the local adapter identifier, such as the worldwide name for Fibre Channel adapters and the current state of the link, if available. The **-r** flag adds information about the remote ports, which are accessed by the adapter.

Resetting statistics

The **-z** flag causes the **lsmpio** command to reset all statistics to zero. By default, the statistics for all devices are reset. The **-l** flag can be used to reset the statistics for only a single device.

Flags

Item	Description
-a	Lists parent Fibre channel adapter information.
-d	Displays detailed statistics. This flag is only valid with the **-S** flag.
-e	When this flag is used with the **-a** flag, error counts for the adapters are displayed. When this flag is used with the **-a** and **-r** flags, error counts for adapters and remote ports are displayed.
-h	Displays command usage information.
-l *disk_name*	Specifies a device. If this flag is included, the command operates on a single device. If this flag is omitted, the command operates on all AIX MPIO devices. This flag can be used by itself for the summary path status, or with the **-q**, **-S**, or **-z** flags.
-q	Queries the device information. This command uses standard SCSI commands to query the device for information. The precise information returned varies, depending on the device type.
-r	Displays the remote port information. This flag is used along with the **-a** flag to display information about the remote ports that are accessed by an adapter. The information that is returned might depend on the protocol that is used by the adapter.
-S	Displays statistics for one or all devices. This flag displays basic counters for path use and path errors. If the **-d** flag is used along with this flag, it displays more detailed statistics.
-z	Resets all statistics. If this flag is used, it causes the PCM to reset all statistical counters back to zero. **Note:** You can use the **-z** flag with the **-a** flag to reset the adapter error counts.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

Examples of displaying MPIO information:

1. To display the summary information for the hdisk1234 device, enter the following command:

   ```
   lsmpio -l hdisk1234
   ```

   The system displays an output similar to the following output:

   ```
   name   path_id status  path_status     parent connection
   ================================================================================
   hdisk1234  0   Enabled Opt,Sel,Deg,Rsv fscsi0 500a098186a7d4ca,0008000000000000
   hdisk1234  1   Enabled Non             fscsi0 500a098196a7d4ca,0008000000000000
   hdisk1234  2   Enabled Opt,Sel         fscsi1 500a098186a7d4ca,0008000000000000
   hdisk1234  3   Enabled Non             fscsi1 500a098196a7d4ca,0008000000000000
   ```

2. To display detailed device statistics for the hdisk10 device, enter the following command:

   ```
   lsmpio -Sdl hdisk10
   ```

   The system displays an output similar to the following output:

   ```
   Disk: hdisk10
       Path statistics since Tue May 21 17:38:43 CDT 2013
       Path 0: (fscsi0:500a098186a7d4ca,8000000000000)
           Path Selections:                                    0
           Adapter Errors:                                     0
               Software:                       0
               Hardware:                       0
               Transport Dead:                 0
               Transport Busy:                 0
               Transport Fault:                0
               No Device Response:             0
               Target Port ID Changed:         0
           Command Timeouts:                                   0
           Reservation Conflicts:                              0
           SCSI Queue Full:                                    0
           SCSI Busy:                                          0
   ```

```
        SCSI ACA Active:                                0
        SCSI Task Aborted:                              0
        SCSI Aborted Command:                           0
        SCSI Check Condition:                           0
            Medium Error:                   0
            Hardware Error:                 0
            Not Ready:                      0
            Other:                          0
        Last Error:                                   N/A
        Last Error Time:                              N/A
        Path Failure Count:                             0
            Due to Adapter Error:           0
            Due to I/O Error:               0
            Due to Health Check:            0
            Due to SCSI Sense:              0
            Due to Qualifier Bit:           0
            Due to Opening Error:           0
        Last Path Failure:                            N/A
        Last Path Failure Time:                       N/A
```

Note: If some SCSI error counts are reported, it does not indicate a problem or that I/O operations have failed. It is normal during regular processing for temporary, recoverable errors to be reported, and for the I/O operation to be attempted again.

3. To display MPIO adapter information with remote port information, enter the following command:

   ```
   lsmpio -ar
   ```

   The system displays an output similar to the following output:

```
Adapter Driver: fscsi0 - AIX PCM
    Adapter WWPN:  10000000c94c7bd6
    Link State:    Up
                         Paths      Paths      Paths      Paths
    Remote Ports        Enabled   Disabled     Failed    Missing         ID
    500a098186a7d4ca         31          0          0          0    0x20a00
    500a098196a7d4ca         31          0          0          0    0x20b00
    500507630a18016b         19          0          0          0    0x31200
    500507630a18416b         19          0          0          0    0x31300
    500507630a18816b         19          0          0          0    0x31400
    500507630a18c16b         19          0          0          0    0x31500

Adapter Driver: fscsi1 - AIX PCM
    Adapter WWPN:  10000000c94c7bd7
    Link State:    Up
                         Paths      Paths      Paths      Paths
    Remote Ports        Enabled   Disabled     Failed    Missing         ID
    500a098186a7d4ca         31          0          0          0    0x20a00
    500a098196a7d4ca         31          0          0          0    0x20b00
    500507630a18016b         19          0          0          0    0x31200
    500507630a18416b         19          0          0          0    0x31300
    500507630a18816b         19          0          0          0    0x31400
    500507630a18c16b         19          0          0          0    0x31500
    5001738000330150          1          0          0          0    0x10100
    5001738000330162          1          0          0          0    0x10200
```

4. To query a device and display information about it, enter the following command:

   ```
   lsmpio -ql hdisk48
   ```

   The system displays an output similar to the following output:

```
Device:  hdisk48
          Vendor Id:  IBM
         Product Id:  2107900
           Revision:  .160
           Capacity:  10G
      Volume Serial:  600507630AFFC16B0000000000001505  (Page 83 NAA)
```

Note: The information that is displayed comes from the standard inquiry data and the device identification VPD data. If the device represents a PPRC pair (the **san_rep_device** attribute has a value of *yes*), the display includes the volume serial number for each LUN in the pair, as well as the vendor-specific ID that is shared by the two LUNS of the pair, as shown here:

```
Device:  hdisk33
          Vendor Id:  IBM
         Product Id:  2107900
           Revision:  .160
           Capacity:  10G
      Volume Serial:  600507630AFFC16B0000000000000113  (Page 83 NAA)
      Volume Serial:  600507630AFFC16B000000000000031F  (Page 83 NAA)
      Vendor LUN Id:  3735544C37373130313133005022AD6A
```

5. To display local and remote port error counts, enter the following command:

```
lsmpio -are
```

The output might be similar to the following sample:

```
Adapter Driver: fscsi3 -> AIX PCM
    Adapter WWPN:  21000024ff6aee7d
    Link State:    Up
    Connectivity Errors:
        Last 10 Minutes:      74
        Last 60 Minutes:     222
        Last 24 Hours:     12345

                                Connectivity Errors
                              Last 10   Last 60   Last 24
    Remote Ports              Minutes   Minutes   Hours
    5001738000330171              0         0         0
    5001738000330173              0         0         0
    500a098286a7d4ca              2         9        45
    500a098196a7d4ca             72       213     12300
```

Related information:

devrsrv command

fcstat command

lspath command

mpio_get_config 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 System Management Interface Tool (SMIT) **smit lsnamerslv** fast path to run this command.

Flags

Item	Description
-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:

namerslv command

Naming command

SMIT Interface for TCP/IP

---

lsnfsexp Command

Purpose

Displays the characteristics of directories that are exported with the Network File System (NFS).

Syntax

/usr/sbin/lsnfsexp [**-c** | **-l**] [*Directory*] [**-V** *Exported Version*] [**-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

Item	Description
-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**.
-V *Exported Version*	Specifies the version of the directory to be displayed. Valid version numbers are 2, 3 and 4.

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
   ```
3. To list the entry for the **/common/documents** directory that is exported as version 4, enter the following command:
   ```
   lsnfsexp /common/documents -V 4
   ```

File

Item	Description
/etc/exports	Lists the directories the server can export.

Related reference:

"mknfsexp Command" on page 752

Related information:

exportfs command

Network File System (NFS) Overview for System Management

Reference command

lsnfsmnt Command

Purpose

Displays the characteristics of NFS mountable file systems.

Syntax

/usr/sbin/lsnfsmnt [**-c** | **-l** | **-p**] [*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

Item	Description
-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.
-p	Specifies that the output should be in pipe format.

Examples

To list all of the NFS mounted file systems in the colon format, enter:

```
lsnfsmnt -c
```

Files

Item	Description
/etc/filesystems	Centralizes file system characteristics.

Related reference:
"mknfsmnt Command" on page 755
Related information:
chnfsmnt command
Network File System
Overview for System Management

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**] | **-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**] | **-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

Item	Description
-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	Is used by the SMIT interface of a NIM environment.
-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

Item	Description
/etc/niminfo	Contains variables used by NIM.

Related information:
nim command
nimclient command
nimconfig command
niminit command
.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:
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 the following ways.

- Use the **-C** flag to display the default output information for a device from the Customized Devices object class, which is name, state, location, and description.
- Use the **-P** flag to display the default output information for a device from the Predefined Devices object class, which is class, type, subclass, and description.

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 using the **-f** flag.

Flags

Item	Description
-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 necessary 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 non-alphanumeric characters. If white space is used 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 the following:

   ```
   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 the following:

   ```
   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 `rmt0` tape device as a child device, type the following:

   ```
   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 `rmt0` tape device as a child device, type the following:

   ```
   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

Item	Description
/usr/sbin/lsparent	Contains the **lsparent** command.

Related reference:

"lsattr Command" on page 354

"lsdev Command" on page 396

"mkdev Command" on page 701

Related information:

chdev command

lspath Command

Purpose

Displays information about paths to an MultiPath I/O (MPIO) capable device.

Syntax

lspath [**-F** *Format* | **–t**] [**-H**] [**-l** *Name*] [**-p** *Parent*] [**-s** *Status*] [**-w** *Connection*] [**-i** *PathID*]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] [**-i** *PathID*] {**-D** [**-O**] | **-E** [**-O**] | **-F** *Format* [**-Z** *character*] } [**-a** *Attribute*] ...[**-f** *File*] [**-H**]

lspath -A -l *Name* **-p** *Parent* [**-w** *Connection*] [**-i** *PathID*]**-R -a** *Attribute* [**-f** *File*] [**-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.

  Note: The AIX MPIO does not fail the last path. Even though all paths to the storage are lost, the last path displays the status as **enabled**. In such a case, an IO error indicates the actual loss of the last path to the disk.

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.

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

Item	Description
-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.

Item	Description
-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.
-i *PathID*	Indicates the path ID associated with the path to be displayed.
-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, whose paths are to be displayed. This flag is optional for displaying path status, but it 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**. You can use alternative forms of the values. Strings beginning with **0**, **d**, or **D** are treated in the same way as **defined**. The only exception is that strings starting with **di**, **Di**, **dI**, or **DI** are treated in the same way as **disabled**. Strings beginning with **1**, **a**, or **A** are treated in the same way as **available**. Strings beginning with **3**, **m**, or **M** are treated in the same way as **missing**. Strings beginning with **e** or **E** are treated in the same way as **enabled**. Strings beginning with **f** or **F** are treated in the same way as **failed**.
-t	Displays the path ID in addition to the current default output. The **-t** flag cannot be used with the **-F** or the **-A** flags.
-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 embedded new line characters. 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 the following command:

   ```
   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 the following command:

   ```
   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 the following command:

   ```
   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 the following command:

   ```
   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, enter the following 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 reference:

"lsmpio Command" on page 461

"mkpath Command" on page 764

lspriv Command

Purpose

Displays the privileges available on the system.

Syntax

lspriv [**-v**]

Description

The **lspriv** command displays privileges available to the system. If run within a workload partition (WPAR), the **lspriv** command displays only the privileges available to the partition. If the **-v** flag is specified, the **lspriv** command also displays privilege descriptions.

Flags

Item	Description
-v	Specifies to display the privilege description for each privilege.

Security

Any user can run the **lspriv** command on the system.

Examples

1. To display all privileges available on the system, use the following command:

   ```
   lspriv
   ```

   The system displays an output similar to the following example:

   ```
   PV_ROOT
   PV_AU_
   PV_AU_ADD
   PV_AU_ADMIN
   PV_AU_READ
   ...
   ```

2. To display all privileges available on the system and their textual description, use the following command:

   ```
   lspriv -v
   ```

   The system displays an output similar to the following example:

   ```
   PV_ROOT Allows a process to pass any non-SU privilege check.
   PV_AU_ Equivalent to all Auditing privileges (PV_AU_*) combined.
   PV_AU_ADD Allows a process to record/add an audit record.
   PV_AU_ADMIN Allows a process to configure and query the audit system.
   PV_AU_READ Allows a process to read a file marked as an audit file.
   ...
   ```

Related information:

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

Item	Description
-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:

chquedev command

ruser command

qdaemon command

Transmission Control Protocol / Internet Protocol

lsps Command

Purpose

Displays the characteristics of a paging space.

Syntax

lsps { **-s** | [**-c** | **-l**] { **-a** | **-t** { *lv* | *nfs* | *ps_helper*} | *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.

If the **-t** flag is specified, the argument will be assumed to be a third-party helper executable. If the helper executable is present in the /sbin/helpers/pagespace path then it will be spawned passing all the arguments and with the **-l** flag to specify the **lsps** command. The helper executable must take care of displaying the characteristics of the page space. If the helper program doesn't exist in /sbin/helpers/pagespace directory, the **lsps** command will display the usage error. The helper executable must exit with a 0 if successful and a non-zero if it fails.

You can use the System Management Interface Tool (SMIT) **smit lsps** fast path to run this command.

Flags

Item	Description
-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:** There is a paging space limit of 64 GB per device. **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. *ps_helper* Name of the helper program for a third party device.

Examples

1. 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   PhysicalVolume  Volume Group    Size   %Used  Active  Auto  Type  Chksum
   hd6           hdisk0          rootvg         512MB    1     yes     yes   lv     8
   ```

2. To display the characteristics of paging space myps using the helper program foo enter the following commad:

   ```
   lsps -t foo myps
   ```

   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
myps             mydisk            myvg            512MB     1    yes   yes    lv
```

Files

Item	Description
/etc/swapspaces	Specifies the paging space devices and their attributes.

Related reference:

"mkps Command" on page 778

Related information:

swap command

swapon command

File systems

---

lspv Command

Purpose

Displays information about a physical volume within a volume group.

Syntax

lspv

OR

lspv [**-L**] [**-P**] [**-l** | **-p** | **-M**] [**-n** *descriptorphysicalvolume*] [**-v** *volumegroupid*] *physicalvolume*

Description

The **lspv** command displays the information about the physical volume if the specific physical volume name is specified. If you do not add flags to the **lspv** command, by default all the available physical volumes are printed along with the following information:

- Physical disk name.
- Physical volume identifiers (PVIDs).
- The volume group, if any, that the physical volume belongs to or the label, if any, locked with the **lkdev** command.
- The state of the volume group.

  Active When the volume group is varied on.

  Concurrent
  When the volume group is varied on in the concurrent mode.

  Locked
  When the physical volume is locked with the **lkdev** command.

Note: If the **lspv** command cannot find the 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 value might be displayed:

```
PP RANGE: ?
```

Note: The **lspv** command, without any flags, can display the General Parallel File System (GPFS) volume groups that are located on the disks. However, the **lspv** command must initially be run with a root

authority so that the command has permissions to query the GPFS nodes for information. After the GPFS volume group names are cached locally, non-root users running the **lspv** command can see the GPFS volume group names.

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:

Item	Description
`Physical volume`	The name of the physical volume.
`Volume group`	The 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`	The 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`	The allocation permission for this physical volume.
`Logical volumes`	The number of logical volumes using the physical volume.
`Stale PPs`	The number of physical partitions on the physical volume that are not current.
`VG descriptors`	The number of volume group descriptors on the physical volume.
`PP size`	The size of physical partitions on the volume.
`Total PPs`	The total number of physical partitions on the physical volume.
`Free PPs`	The number of free physical partitions on the physical volume.
`Used PPs`	The number of used physical partitions on the physical volume.
`Max Request`	The max transfer size of the physical volume.
`Free distribution`	The number of free partitions available in each intra-physical volume section.
`Used distribution`	The number of used partitions in each intra-physical volume section.
`Mirror Pool`	The mirror pool that the physical volume has been assigned to.

You can use the System Management Interface Tool (SMIT) **smit lspv** fast path to run this command.

Flags

Item	Description
-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.

Item	Description
-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.
-P	Lists the mirror pool that each physical volume belongs to.
-u	Lists all the physical volumes in the system along with the following information: • Physical disk name. • Physical volume identifiers (PVIDs). • The volume group (if any), or label (if any), that the physical volume belongs to and that is locked with the **lkdev** command. • The state of the volume group. **Active** When the volume group is varied on. **Concurrent** When the volume group is varied on in the concurrent mode. **Locked** When the physical volume is locked with the **lkdev** command. • Unique device identifier (UDID). • Universally Unique Identifier (UUID).
-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 the following command:
   ```
   lspv hdisk3
   ```
2. To display the status and characteristics of physical volume `hdisk5` by physical partition number, enter the following command:

```
lspv  -p hdisk5
```

Files

Item	Description
/usr/sbin	Contains the **lspv** command.

Related reference:
"lslv Command" on page 452
Related information:
mkvg command
varyonvg command
varyoffvg command
Logical volume storage

lspprc Command

Purpose

Displays information about PPRC (Peer-to-Peer Remote Copy) disks.

Syntax

lspprc -A *[-o]*

lspprc -c *pprc_disk*

lspprc *[-h]*

lspprc -p *pprc_disk*

lspprc -v *pprc_disk*

Description

The **lspprc** command displays information that is related to PPRC disk like vital product data (VPD) information of the individual LUNs that are part of the PPRC disk, path group information of a PPRC disk, replication path information of a PPRC disk, and a list of all PPRC disks available on the system.

Flags

Item	Description
-A *[-o]*	Displays the information of all PPRC disks in the system, such as PPRC state and path groups IDs. The optional **-o** flag briefly opens all potential PPRC disks before displaying the status, ensuring that the status is current. Use of the **-o** flag might increase the length of time that is required to run the command, depending on the number of disks that are attached to the system.

Example:

```
hdisk#   PPRC      Primary        Secondary    Primary Storage   Secondary Storage
         state     path group     path group   WWNN              WWNN
                   ID             ID
hdisk4   Active    0(s)            1           500507630affc16b  500507630affc16b
hdisk5   Active    0(s)           1            500507630affc16b  500507630affc16b
hdisk6   Active    0(s)           1            500507630affc16b  500507630affc16b
```

Note: For explanation of the path group IDs and the selected path group that is identified as *s*, see the description of the **-p** flag.

Item	Description

-c	Displays information about the replication path connection, which is related to a PPRC disk. The output displays information about the replication paths between the two storage subsystems that contain the LUNs in the PPRC pair. The output displays the worldwide node names, Subsystem IDs (SSID), Logical Subsystems (LSS), and ports for the endpoints of the paths and the current path state. The contents of the output depends on the current state of the PPRC disk.
	• If the disk is not part of a PPRC pair, the output shows all paths that originate from the LSS that the disk resides on, to any other LSS.
	• If the disk is part of a PPRC pair, but has only one path group, the output displays paths that originate from the LSS on which the disk resides and lead to the LSS on which the partner disk resides.
	• If the disk is part of a PPRC pair and has two path groups, the output displays all paths between the two LSSs on which the members of the PPRC pair reside.
	
	Example output of the replication paths:
	`lspprc -c hdisk33`
	
	`Displays all paths between LSS 01 and LSS 03`
	
	`Source Target`
	`WWNN SSID LSS Port WWNN SSID LSS Port State`
	`==========`
	`500507630AFFC16B FF03 03 0301 500507630AFFC16B FF01 01 0302 Up`
	`500507630AFFC16B FF03 03 0302 500507630AFFC16B FF01 01 0303 Up`
	`500507630AFFC16B FF01 01 0300 500507630AFFC16B FF03 03 0302 Up`

-p	Displays path group information that is part of the specified PPRC disk.
	
	Example output of path group information:
	`lspprc -p hdisk55`
	`Path WWNN LSS VOL Path`
	`group id group status`
	`==========`
	`0(s) 500507630affc16b 0xf 0x1c PRIMARY`
	`1 5005076303ffd2ea 0xc1 0x0 SECONDARY`
	
	`path path path parent connection`
	`group id id status`
	`==========`
	`0 0 Available fscsi0 500507630a08016b,400f401c00000000`
	`0 1 Available fscsi0 500507630a08416b,400f401c00000000`
	`1 2 Available fscsi1 50050763030812ea,40c1400000000000`
	`1 3 Available fscsi1 50050763030852ea,40c1400000000000`
	
	When you use PPRC, the paths are grouped based on which LUN is accessed by the path, in the PPRC pair. The path group ID indicates how the paths are grouped, with all paths that have the same path group ID accessing the same LUN in the PPRC pair. A path group ID of -1 indicates that there are no paths that are configured from this initiator to the indicated LUN in the PPRC pair.
	
	At any time, only one of the two path groups is selected for I/O operations to the hdisk. The selected path group is identified in the output by "(s)".

Item	Description

-v	Displays the VPD information of individual LUNs that are part of given PPRC disk.
	
	Example output of VPD information:
	`# lspprc -v hdisk0`
	
	`Hyperswap lun unique identifier.....35203735544c3737313`
	`037303000502a14ae07210790003IBMfcp`
	
	`hdisk0  Primary         MPIO IBM 2107 FC Disk`
	
	`Manufacturer.....IBM`
	`Machine Type and Model.....2107900`
	`ROS Level and ID.....2E313630`
	`Serial Number.....75TL7710`
	`Device Specific.(Z7).....0700`
	`Device Specific.(Z0).....000005329F101002`
	`Device Specific.(Z1).....700`
	`Device Specific.(Z2).....075`
	`Unique Device Identifier.....200B75TL771070007210790003IBMfcp`
	`Logical Subsystem ID.....0x07`
	`Volume Identifier.....0x00`
	`Subsystem Identifier(SS ID)...0xFF07`
	`Control Unit Sequence Number..00000TL771`
	`Storage Subsystem WWNN.....500507630affc16b`
	`Logical Unit Number ID.....4007400000000000`
	
	`hdisk0  Secondary       MPIO IBM 2107 FC Disk`
	
	`Manufacturer.....IBM`
	`Machine Type and Model.....2107900`
	`ROS Level and ID.....2E313630`
	`Serial Number.....75TL7710`
	`Device Specific.(Z7).....0900`
	`Device Specific.(Z0).....000005329F101002`
	`Device Specific.(Z1).....900`
	`Device Specific.(Z2).....075`
	`Unique Device Identifier.....200B75TL771090007210790003IBMfcp`
	`Logical Subsystem ID.....0x09`
	`Volume Identifier.....0x00`
	`Subsystem Identifier(SS ID)...0xFF09`
	`Control Unit Sequence Number..00000TL771`
	`Storage Subsystem WWNN.....500507630affc16b`
	`Logical Unit Number ID.....4009400000000000`

Files

Item	Description
/usr/bin/lspprc	Contains the **lspprc** command.

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

Item	Description
-c	Causes colon output format for use by SMIT.
-q *Name*	Specifies the *Name* of the queue stanza that is sent to standard output.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/lsque	Contains the **lsque** command.
/etc/qconfig	Contains the configuration file.

Related reference:

"lsquedev Command"

Related information:

qconfig file

Printer-specific information

printf subroutine

---

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/lsquedev	Contains the **lsquedev** command.
/etc/qconfig	Contains the configuration file.

Related reference:

"mkquedev Command" on page 782

Related information:

qconfig command

Installing support for additional printers

Virtual printer definitions and attributes

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:

Item	Description
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:

Item	Description
A	Indicates an AT interrupt controller.
B	Indicates a non-AT interrupt controller.

Flags

Item	Description
-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

Item	Description
/usr/sbin/lsresource	Contains the **lsresource** command.

lsresponse Command

Purpose

Lists information about one or more responses.

Syntax

lsresponse [**-a**] [**-C** | **-l** | **-t** | **-d** | **-D** *delimiter*] [**-A**] [**-q**] [**-U**] [**-x**] [**-b**] [**-h**] [**-TV**] [*response1*[**,***response2*,...] **:***node_name*]

Description

The **lsresponse** command lists the following information about defined responses:

Field	Description
ResponseName	The name of the response.
Node	The location of the response.
Action	The name of an action.
DaysOfWeek	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** values are separated by commas (**,**). The number of **DaysOfWeek** values must match the number of **TimeOfDay** values. The values for each day follow: **1** Sunday **2** Monday **3** Tuesday **4** Wednesday **5** Thursday **6** Friday **7** Saturday
TimeOfDay	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. Multiple **TimeOfDay** values are separated by commas (**,**). The number of **DaysOfWeek** values must match the number of **TimeOfDay** values.
ActionScript	The script or command to run for the action.
ReturnCode	The expected return code for **ActionScript**.
CheckReturnCode	Indicates whether the actual return code for **ActionScript** is compared to its expected return code. The values are: **y** (yes) and **n** (no).
EventType	The type of event that causes the action to be run: event, rearm event, or both.
StandardOut	Indicates whether standard output is directed to the audit log. The values are: **y** (yes) and **n** (no).
EnvironmentVars	Indicates any environment variables that will be set before the action is run.
UndefRes	Indicates whether the action is to be run if a monitored resource becomes undefined. The values are: **y** (yes) and **n** (no).
Locked	Indicates whether the resource is locked or unlocked.
EventBatching	Indicates whether the response action supports event batching.

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.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

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.

-A Displays all of the attributes of the response.

-b Displays only the responses that support event batching.

-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.

-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.

-l Displays the response information on separate lines (long form).

-q Does not return an error when **response** does not exist.

-t Displays the response information in separate columns (table form).

-U Indicates whether the resource is locked.

-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 *Administering RSCT* 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_IP_AUTHENT
: When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
: Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

  0 Specifies *local* scope.

  1 Specifies *local* scope.

  2 Specifies *peer domain* scope.

  3 Specifies *management domain* scope.

  If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list all of the responses, run this command:

   ```
   lsresponse
   ```

   The output will look like this:

   ```
   ResponseName
   "E-mail root anytime"
   "E-mail root first shift"
   "Critical notifications"
   "Generate SNMP trap"
   ```

2. To see which resources are locked, run this command:

   ```
   lsresponse -U
   ```

   The output will look like this:

   ```
   ResponseName                   Node      Locked
   "Broadcast event on-shift"     "nodeA"   "No"
   "E-mail root off-shift"        "nodeA"   "No"
   "E-mail root anytime"          "nodeA"   "No"
   "Log event anytime"            "nodeA"   "No"
   "Informational notifications"  "nodeA"   "No"
   "Warning notifications"        "nodeA"   "No"
   "Critical notifications"       "nodeA"   "No"
   "Generate SNMP trap"           "nodeA"   "No"
   ```

3. To list general information about the response "Critical notifications", run this command:

   ```
   lsresponse "Critical notifications"
   ```

   The output will look like this:

   ```
   ResponseName     = "Critical notifications"
   Node             = "nodeA"
   Action           = "Log Critical Event"
   DaysOfWeek       = 1+2+7
   TimeOfDay        = 0000-2400
   ActionScript     = "/opt/rsct/bin/logevent /tmp/criticalEvents"
   ReturnCode       = 0
   CheckReturnCode  = "y"
   EventType        = "b"
   StandardOut      = "y"
   EnvironmentVars  = '"Env1=5","Env=10"'
   UndefRes         = "n"

   ResponseName     = "Critical notifications"
   Node             = "nodeA"
   Action           = "E-mail root"
   DaysOfWeek       = 6+2,6+2,6+5
   TimeOfDay        = 1700-2400,0000-0800,0000-2400
   ActionScript     = "/opt/rsct/bin/notifyevent root"
   ```

```
ReturnCode       = 0
CheckReturnCode  = "y"
EventType        = "b"
StandardOut      = "y"
EnvironmentVars  = ""
UndefRes         = "n"
```

4. 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"
   ```

5. 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:

   ```
   ResponseName = "E-mail root anytime"
   Action       = "E-mail root"
   ⋮
   ResponseName = "E-mail root first shift"
   Action       = "E-mail root"
   ```

Location

/opt/rsct/bin/lsresponse

---

lsrole Command

Purpose

Displays role attributes.

Syntax

lsrole [**-R** *load_module*] [**-c** | **-f** | **-C**] [**-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 System Management Interface Tool (SMIT) **smit lsrole** fast path to run this command.

If the system is configured to use multiple domains for the role database, the roles, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the roles stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the roles from a specific domain.

The **lsrole** command only lists the role definitions available in the roles database. If the system is operating in enhanced Role Based Access Control (RBAC) mode, the information in the roles database might differ from what is used for security considerations on the system in the kernel security tables (KST). To view the state of the roles database in the KST, use the **lskst** command.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable can include any attribute that is defined in the **chrole** command. Specify more than one attributes with a blank space between attribute names. If an empty list is specified, only the role names are displayed. In addition to the attributes defined in the **chrole** command, the following attributes can also be listed with the **-a** flag: **all_auths** Traverses the role hierarchy of the specified roles and gathers all the authorizations. The **all_auths** attribute differs from the **authorizations** attribute because the **lsrole** command only lists the explicit authorizations of the specified roles for that attribute. **users** Displays the users that are granted the specified roles. **description** Displays the text description of the role as indicated by the **dfltmsg**, **msgcat**, **msgset** and **msgnum** attributes for the role.

-c	Displays the role attributes in colon-separated records, as follows:
	`# role: attribute1: attribute2: ...`
	`Role: value1: value2: ...`

-C	Displays the role attributes in colon-separated records that are easier to parse than the output of the **-c** flag:
	`#role:attribute1:attribute2: ...`
	`role:value1:value2: ...`
	`role2:value1:value2: ...`
	The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you specified the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If a role does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.

-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`

-R *load_module*	Specifies the loadable module to list roles from.

Security

The **lsrole** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.role.list	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
r	/etc/security/roles

Examples

1. To display the role `rolelist` and groups of the role `ManageAllUsers` in a colon format, use the following command:

   ```
   lsrole -c -a rolelist groups ManageAllUsers
   ```

   Information similar to the following appears:

   ```
   # role: rolelist:groups
    ManageAllUsers: ManagerBasicUser:security
   ```

2. To list all attributes of the `ManageAllUsers` role from LDAP, use the following command:

   ```
   lsrole -R LDAP ManageAllUsers
   ```

   All the attribute information appears, with each attribute separated by a blank space.

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.

Related reference:

"mkrole Command" on page 790

"mkuser Command" on page 839

Related information:

setkst command

Security

---

lsrpdomain Command

Purpose

Displays peer domain information for the node.

Syntax

lsrpdomain [**-o** | **-O**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*peer_domain*]

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 the name of a peer domain, 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:

Field	Description
Name	The name of the peer domain.
RSCTActiveVersion	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	The topology services port number.
GSPort	The group services port number.
OpState	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.

-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

peer_domain
Specifies the name of the peer domain about which you want to display information. You can specify a peer domain name or a substring of a peer domain name for this parameter. If you specify a substring, the command displays information about any defined peer 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.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on the node for which the peer domain information is requested.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To display general information about the peer domains to which **nodeA** belongs, run this command on **nodeA**:

   ```
   lsrpdomain
   ```

   The output will look like this:

   ```
   Name            OpState         RSCTActiveVersion       MixedVersions   TSPort  GSPort
   ApplDomain      Online          2.5.0.0                  No              12347   12348
   ```

2. To display general information about the peer domains to which **nodeA** belongs, with the default delimiter (but without the heading), run this command on **nodeA**:

   ```
   lsrpdomain -xd
   ```

   The output will look like this:

   ```
   ApplDomain:Online:2.5.0.0:No:12347:12348:
   ```

3. To display general information about the peer domains to which **nodeA** belongs, in long format, run this command on **nodeA**:

```
lsrpdomain -l
```

The output will look like this:

```
Name              = ApplDomain
OpState           = Online
RSCTActiveVersion = 2.5.0.0
MixedVersions     = No
TSPort            = 12347
GSPort            = 12348
```

Location

/opt/rsct/bin/lsrpdomain

---

lsrpnode Command

Purpose

Displays information about one or more of the nodes that are defined in the online peer domain.

Syntax

lsrpnode [[**-i**] [**-l** | **-t** | **-d** | **-D** *delimiter*] **-o** | **-O** | **-L**] **-P**] **-Q**] [**-B**] [**-x**] [**-h**] [**-TV**] [*node_name*]

lsrpnode -p *peer_domain* [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-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**, **-O**, and **-L** flags also limit the information this command displays. The **-o** flag displays information about nodes that are online. The **-O** flag displays information about nodes that are offline. The **-L** flag displays information about the local node, which is the node the command runs on.

The **-P** flag displays additional node configuration information related to group services group leader selection. The **-Q** flag displays additional node configuration information related to quorum decisions. The **-B** flag displays additional node configuration information related to the tiebreaker mechanism.

By default, the **lsrpnode** command displays information in table format (**-t**).

Some of the node information that is displayed follows:

Field	Description
Name	The name of the node in the peer domain.
OpState	The operational state of the node.
RSCTVersion	The version of RSCT that is active in the node.

The following fields are displayed when you specify the **-i** flag:

Field	Description
NodeNum	The node number used by topology services and group services. This number is unique within the cluster.
NodeID	The unique node identifier.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-P** flag:

Field	Description
Preferred	Indicates whether the node is a group services group leader candidate.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-Q** flag:

Field	Description
Quorum	Indicates whether the node participates in quorum decisions.

Along with other fields (depending on the flags specified), this field is displayed when you specify the **-B** flag:

Field	Description
Tiebreaker	Indicates whether the node has access to the peer domain's tiebreaker mechanism.

See the *Administering RSCT* guide for information about group services group leader selection, quorum decisions, and the tiebreaker mechanism.

Flags

-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.

-i Displays the node number and node ID for the node. The node number is used by topology services and group services and is unique within the cluster. The node ID is the unique node identifier.

-l Displays the information on separate lines (long format).

-L Displays information about the local node only, which is the node that the command runs on.

-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.

-p *peer_domain*
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. You must have root access to use the **-p** flag.

-P Indicates whether the node is a group services group leader candidate. **yes** is displayed if the node can be a group services group leader. **no** is displayed if the node cannot be a group services group leader. See the *Administering RSCT* for more information about group services group leader selection.

-Q Indicates whether the node participates in quorum decisions. yes is displayed if the node

participates in quorum decisions. no is displayed if the node does not participate in quorum decisions. See the *Administering RSCT* for more information on quorum decisions.

-B Indicates whether the node has access to the peer domain's tiebreaker mechanism. yes is displayed if the node has access to the peer domain's tiebreaker mechanism. no is displayed if the node does not have access to the peer domain's tiebreaker mechanism. See the *Administering RSCT* for more information on the tiebreaker mechanism.

-t Displays the information in separate columns (table format). This is the default format.

-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

node_name
Specifies the name of the node about which you want to display information. 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.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

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    3.1.4.0
   nodeB   Online    3.1.4.0
   nodeC   Offline   3.1.4.0
   ```

2. To display general information about the nodes in the online peer domain that **nodeA** belongs to, with the default delimiter (but without the heading), run this command on **nodeA**:

   ```
   lsrpnode -xd
   ```

   The output will look like this:

   ```
   nodeA:Online:3.1.4.0:
   nodeB:Online:3.1.4.0:
   nodeC:Offline:3.1.4.0:
   ```

3. To display general information about the nodes in the online peer domain that **nodeA** belongs to, in long format, run this command on **nodeA**:

   ```
   lsrpnode -l
   ```

   The output will look like this:

   ```
   Name        = nodeA
   OpState     = Online
   RSCTVersion = 3.1.4.0

   Name        = nodeB
   OpState     = Online
   RSCTVersion = 3.1.4.0

   Name        = nodeC
   OpState     = Offline
   RSCTVersion = 3.1.4.0
   ```

4. To display general information about the nodes in the online peer domain that **nodeA** belongs to, including the node number and node ID, run this command on **nodeA**:

   ```
   lsrpnode -i
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   NodeNum   NodeID
   nodeA   Online    3.1.4.0       2         40a514bed9d82412
   nodeB   Online    3.1.4.0       1         47fe57098f4ec4d9
   ```

5. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the preferred group services group leader information, run this command on **nodeA**:

   ```
   lsrpnode -P
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Preferred
   nodeA   Online    3.1.4.0       yes
   nodeB   Online    3.1.4.0       no
   ```

6. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the quorum information, run this command on **nodeA**:

   ```
   lsrpnode -Q
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Quorum
   nodeA   Online    3.1.4.0       no
   nodeB   Online    3.1.4.0       yes
   nodeC   Online    3.1.4.0       yes
   ```

7. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including quorum and tiebreaker information, run this command on **nodeA**:

   ```
   lsrpnode -QB
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Quorum  Tiebreaker
   nodeA   Online    3.1.4.0       no    no
   nodeB   Online    3.1.4.0       yes   yes
   nodeC   Online    3.1.4.0       yes   yes
   ```

Location

/opt/rsct/bin/lsrpnode

---

lsrset Command

Purpose

Displays system rset contents.

Syntax

lsrset [**-X**] [-f] [-v | -o] [[**-S**] -r *rsetname* | -n *namespace* | -a]

or

lsrset [**-X**] [-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

Item	Description
-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.
-S	Displays the resources contained in this rset if it were to be scheduled with the **-S** hint with either the **execrset** or the **attachrset** command. The rset does not need to be an exclusive rset. This is to be contrasted with the **attachrset** and **execrset** commands, which require exclusive rsets to be specified with the **-S** flag.
-X	Prints all available characters of each user and group name instead of truncating to the first 8 characters.

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
   ```
6. To display all resources for all rsets in the system registry with expanded user and group name, type:
   ```
   lsrset  -X -v -a
   ```

Files

Item	Description
/etc/bin/lsrset	Contains the **lsrset** command

Related information:

attachrset command

detachrset command

execrset command

---

lsrsrc Command

Purpose

Displays attributes and values for a resource or a resource class.

Syntax

To display the attributes and values for a *resource:*

lsrsrc [**-s** *"selection_string"*] [**-a** | **-N** { *node_file* | **"-"** }] [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*resource_class*] [*attr...*]

lsrsrc -r [**-s** *"selection_string"*] [**-a** | **-N** { *node_file* | **"-"** }] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*resource_class*]

To display the attributes and values for a *resource class:*

lsrsrc -c [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-a**] [**-h**] [**-TV**] *resource_class* [*attr...*]

lsrsrc -C *domain_name...* [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr...*]

To display a list of all of the resource classes:

lsrsrc

Description

The **lsrsrc** command displays the persistent and dynamic attributes and their values for a resource or a resource class.

Instead of specifying multiple node names in *selection_string*, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "-"** to read the node names from standard input.

When one or more attribute names are specified, these names and their values are displayed in the order specified, provided that each of the specified attribute names is valid. When no attribute names are specified:

- the **-A p** | **d** | **b** flag controls whether persistent attributes or dynamic attributes or both — and their values — are displayed.
- only attributes that are defined as **public** are displayed. Use the **-p** flag to override this default.

For best performance, specify either the **-A p** flag or only persistent attributes as parameters.

Specify the **-r** flag to display only the resource handles associated with the resources for the specified resource class.

To display a list of the attributes and values for a resource class, specify the **-c** flag.

By default, the resource attributes and values are displayed in long format. Use the **-t**, **-d**, or **-D** flag to display the resources in table format or delimiter-formatted output.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

The **lsrsrc** command does not list any attributes that have a datatype defined as **ct_none** (**Quantum**, for example). 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. This flag overrides the **-r** flag.

-C *domain_name...*
Displays the class attributes of a globalized resource class on one or more RSCT peer domains that are defined on the management server. Globalized classes are used in peer domains and management domains for resource classes that contain information about the domain. To display class attributes of a globalized resource class on all peer domains defined on the management server, use the **-c** flag with the **-a** flag instead of **-C**. The command returns the name of the peer domain in the form of an attribute **ActivePeerDomain**. This is not an actual attribute, but is presented as such to indicate which peer domain is being displayed.

-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.

-i Generates a template of *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 **lsrsrc** command be directed to a file. This flag overrides the **-s** and **-A d** flags.

-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.

-N { *node_file* **| "-" }**
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*.
- A number sign (#) in column 1 indicates that the line is a comment.
- Any blank characters to the left of a node name are ignored.
- Any characters to the right of a node name are ignored.

Use **-N** "-" to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-p *property*
Displays attributes with the specified *property*. By default, only public attributes are displayed. To display all of 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 reqd_for_define (required)

0x0004 inval_for_define (not valid)

0x0008 option_for_define (optional)

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 "OR"ed value with the **-p** flag. For example, to display attributes and their values for all persistent attributes that are either **reqd_for_define** or **option_for_define**, enter:

```
lsrsrc -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 table format. Each attribute is displayed in a separate column, with 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

resource_class
Specifies the name of the resource class with the resources that you want to display.

attr... Specifies one or more attribute names. 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.

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 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect 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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all of the resource classes, enter:

   ```
   lsrsrc
   ```

   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:

   ```
   lsrsrc -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:

   ```
   lsrsrc -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:

```
lsrsrc 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
```

6. To list the nodes in the cluster that have at least four processors, using the **/tmp/common/node_file** file:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

as input, enter:

```
lsrsrc -s "NumProcessors >= 4" -N /tmp/common/node_file -t IBM.Host \
Name NumProcessors
```

The output will look like this:

```
Resource Persistent Attributes for IBM.Host
Name              NumProcessors
"node1.ibm.com"   4
"node2.ibm.com"   4
```

Location

/opt/rsct/bin/lsrsrc

---

lsrsrcassoc Command

Purpose

Retrieves a list of resources that are associated with a class using an association provider.

Syntax

lsrsrcassoc [**-s "***source_selection_string***"**] [**-c** *association_class*] [**-d** *association_endpoint_class*] [**-S "***destination_selection_string***"**] [**-o** *role*] [**-R** *result_role*] [**-h**] [**-TV**] *source_class_name* [*property_list*...]

Description

You can use the **lsrsrcassoc** command to learn about the relationships among CIM resources.

This command is an interface into the association query mechanism of the Common Information Model (CIM) resource manager. Association providers that are registered with the CIM resource manager are called to retrieve association data. Before using **lsrsrcassoc**, it might be helpful to run the **lsassocmap** command to find out which association classes are known to the resource monitoring and control (RMC) subsystem.

You must specify a source class name with the **lsrsrcassoc** command. With no flags specified, **lsrsrcassoc** retrieves all resources associated with every resource of this class. Flags can be used to filter which associated resources are displayed.

The command output is similar to that of **lsrsrc**. Resources associated with a source resource are displayed with their class name and one attribute per line to facilitate searching and filtering the output.

Parameters

source_class_name
Specifies the source class in the association.

property_list
Specifies one or more property names. Only these properties (or attributes, in RMC terminology) of associated resources are displayed. If you do not specify this parameter, all property names are displayed.

Flags

-s *source_selection_string*
Specifies that only resources of the source class that match the selection string are used in the search for associated resources.

-S *destination_selection_string*
Specifies that only resources of the associated classes that match this selection string are displayed.

-c *association_class*
Limits the association search to only those resources tied to the source class through *association_class*.

-d *association_endpoint*
Limits the search of associated resources to just the members of this class.

-o *role* The CIM association interface defines the *role* parameter as the name of the property referring to the class on the source side of the association. Typical values for this parameter are "GroupComponent" or "PartComponent", though the specific name must come from the association class definition.

-R *result_role*
Used like the **-o** flag, except this is the name of the property that refers to the destination side of the association.

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages are written to standard output.

Standard error

When the **-T** flag is specified, this command trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with the command-line interface (CLI) script.

2 An incorrect flag was specified on the command line.

3 An incorrect parameter was specified on the command line.

4 The source endpoint class was not found.

5 The destination endpoint class was not found.

6 The association class was not found.

Implementation specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack and Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
/opt/rsct/bin/lsrsrcassoc	

Examples

To view instances of **cimv2.IBMAIX_UnixProcess** (for AIX) and **cimv2.Linux_UnixProcess** (for Linux) that are associated with **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem** respectively on the specified node, enter:

For AIX:

```
lsrsrcassoc -c cimv2.IBMAIX_OSProcess -s 'Name=~"c175nf14"' -S \

'Name=~"emacs"' cimv2.IBMAIX_OperatingSystem Handle Parameters
```

For Linux:

```
lsrsrcassoc -c

cimv2.Linux_OSProcess -s 'Name=~"c175nf14"' -S \

'Name=~"emacs"' cimv2.Linux_OperatingSystem Handle Parameters
```

In these examples:

- **-c cimv2.IBMAIX_OSProcess** and **-c cimv2.Linux_OSProcess** are the association classes whose provider is used.
- **-s 'Name=~"c175nf14"'** is the selection string against the **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem** instances (we only want objects associated with the OS instance representing the node **c175nf14**).
- **-S 'Name=~"emacs"'** is the selection string against **cimv2.IBMAIX_UnixProcess** and **cimv2.Linux_UnixProcess** objects; only those with **Name** attributes that contain the pattern **emacs** are returned.
- **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem**, which are the "source object" parameter, are one of the classes in the association.
- **Handle Parameters** are properties that the provider is asked to return. **Handle** is the PID of the process; **Parameters** is a list of arguments to the process.

The following output is displayed:

```
Resource Persistent Attributes for cimv2.IBMAIX_UnixProcess (or cimv2.Linux_UnixProcess)

resource 1:

Handle = "2781"

Parameters = {"emacs", "-u", "foo.C"}

resource 2:

Handle = "2782"

Parameters = {"emacs", "bar.C"}

resource 3:

Handle = "2783"

Parameters = {"emacs","foo_bar.C"}

resource 4:

Handle = "2784"

Parameters = {"emacs","bar_foo.C"}

resource 5:

Handle = "2785"

Parameters = {"emacs","CIMRC.C"}

resource 6:

Handle = "26994"

Parameters = {"emacs","lsassocmap.pl"}
```

lsrsrcdef Command

Purpose

Displays definition information for a resource or a resource class.

Syntax

For a *resource*...

To display the definition:

lsrsrcdef [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the persistent attribute definitions:

lsrsrcdef -A p [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the dynamic attribute definitions:

lsrsrcdef -A d [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

For a *resource class*...

To display the definition:

lsrsrcdef -c [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the persistent attribute definitions:

lsrsrcdef -c -A p [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the dynamic attribute definitions:

lsrsrcdef -c -A d [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display a list of all of the resource class names:

lsrsrcdef

Description

The **lsrsrcdef** command displays the definition of a resource or a resource class or the persistent or dynamic attribute definitions of a resource or a resource class. By default:

- if no *attr* parameters are specified on the command line, this command displays the definitions for **public** attributes. To override this default, use the **-p** flag or specify the name of the attribute you want to display.
- this command does not display attribute descriptions. To display attribute definitions and descriptions, specify the **-e** flag.

Flags

-A p | d
: Specifies the attribute type. You can display either persistent or dynamic attribute definitions. Use this flag with the **-c** flag to display the persistent or dynamic attribute definitions of a resource class.

  p Displays only persistent attributes

  d Displays only dynamic attributes

-c Displays the definition of a resource class definition. To display the persistent attribute definitions

for a resource class, specify this flag with the **-A p** flag. To display the dynamic attribute definitions for a resource class, specify this flag with the **-A d** flag.

-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 Specifies expanded format. By default, the descriptions of the definitons are not displayed. Specify this flag to display the definitions and the descriptions.

-i Generates a template of *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" format — one entry per line. This is the default display format. If the **lsrsrcdef -l** command is issued without a resource class name, this 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 reqd_for_define (required)

0x0004 inval_for_define (not valid)

0x0008 option_for_define (optional)

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 "OR"ed value with the **-p** flag. For example, to request the attribute definitions for all persistent attributes that are either **reqd_for_define** or **option_for_define**, enter:

```
lsrsrcdef -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 table format. Each attribute is displayed in a separate column, with 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

resource_class
Specifies the name of the resource class with the attribute definitions you want to display.

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 of the attributes for the resource are displayed. Specify individual attribute names 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.

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 *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect 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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To display 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 display 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 display 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","inval_for_define"}
        description             = ""
        attribute_id            = 0
        group_id                = 255
        data_type               = "uint32"
        variety_list            = {{1..1}}
        variety_count           = 1
        default_value           = 0
```

4. To display 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         = {"reqd_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","inval_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 display 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

/opt/rsct/bin/lsrsrcdef

---

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**] [**-n**] [**-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, USB disks, or tapes.

The **lssavevg -r** and **restorevgfiles** commands perform identical operations and must be considered interchangeable.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** *block* flag. You might 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 defaults to 100.
-B	Prints the volume group backup log to **stdout**. This flag displays 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 that is used to make backup, date, shrink size, full size of the backup, and recommended maintenance or technology level (if any). **Note:** The shrink size is the size of the data on all file systems. The full size is total size of each file system (unused + data).
-c	Produces colon-separated output. This flag works only with the **-l** and **-L** flags.
-d *path*	Specifies the directory path to which the files are 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.
-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**.

Item	Description
-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 or technology level, backup size in megabytes, and backup shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused + data). The **-l** flag also displays the logical volume and file system 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**.
-n	Does not restore ACLs, PCLs, or extended attributes
-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 are restored. If the **-r** flag is not used, then executing the **lssavevg** command lists only 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

Item	Description
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 must be specified in the space-separated list. All files in the specified directory are 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 that is on the default device **/dev/rmt0**, enter the following command:

   ```
   lssavevg
   ```

2. To list the contents of the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lssavevg -f /dev/cd1
   ```

3. To list the contents of the system backup that is on device **/dev/cd1**, which is a user volume group that is not rootvg, enter the following command:

   ```
   lssavevg -f /dev/cd1 -s
   ```

4. To restore **/etc/filesystems** from the system backup that is on device **/dev/cd1**, enter the following command:

   ```
   lssavevg -f /dev/cd1 -r ./etc/filesystems
   ```

5. To restore all files in the **/myfs/test** directory of the non-rootvg backup, which is on device **/dev/cd1**, and write the restored files to **/data/myfiles**, enter the following command:

   ```
   lssavevg -f /dev/cd1 -r -s -d /data/myfiles ./myfs/test
   ```

6. To display colon-separated lpp information about a **mksysb** backup tape that is on device **/dev/rmt0**, enter the following command:

   ```
   lssavevg -Lc -f /dev/rmt0
   ```

7. To display the volume group backup log to **stdout**, enter the following command:

   ```
   lssavevg -B
   ```

8. To list volume group and general backup data about a backup that is on **/tmp/mybackup**, enter the following command:

   ```
   lssavevg -l -f /tmp/mybackup
   ```

9. To verify the readability of each header on a volume group backup tape in **/dev/rmt0**, enter the following command:

   ```
   lssavevg -V -f /dev/rmt0
   ```

10. To list the contents of the system backup that is on device **/dev/usbms0,** enter the following command:

    ```
    lssavevg –f /dev/usbms0
    ```

Files

Item	Description
/usr/bin/lssavevg	Contains the **lssavevg** command

Related information:

restorevgfiles command

---

lssavewpar Command

Purpose

Lists the contents of a workload partition backup on a specified media.

Syntax

lssavewpar [**-b** *blocks*] [**-f** *device*] [**-a**] [**-c**] [**-D**] [**-l** | **-L** | **-M** | **-N**] [**-V**]

Description

The **lssavewpar** command lists the contents of a workload partition backup from tape, file, CD, USB flash drive, or DVD.

Flags

Item	Description
-a	Verifies the physical block size of the tape backup, as specified by the **-b** flag. You might 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 to read is 100, which is the default. The **-b** flag is valid only when a tape backup is used.
-c	Produces colon-separated output. Specify the **-c** flag only with the **-l** , **-L**, **-M**, and **-N** flags.
-D	Produces the output of debugging.
-f *device*	Specifies the device that contains the backup (file, tape, CD, or other sources) as defined by the *device* parameter. If you do not specify the **-f** flag, the default device name is **/dev/rmt0**.
-l	Displays information about a workload partition backup. Displays information including the volume group, the date and time that the backup was made, the uname output from the backed up system, the operating system level, the maintenance or technology level, the backup size in megabytes, and the backup-shrink size in megabytes. The shrink size is the size of the data on all file systems. The full size is the total size of each file system (unused and data). The **-l** flag also displays the logical volume and file system information of the backed up workload partition. You must specify the **-f** flag when you specify the **-l** flag. The **-l** flag is mutually exclusive with the **-L**, **-M**, and **-N** flags.
-L	Displays lpp file-set information about a workload partition backup only. When you specify the **-L** flag, you must also specify the **-f** device flag. The **-L** flag is mutually exclusive with the **-l**, **-M**, and **-N** flags.

Item	Description
-M	Lists information about any writable namefs-mounted file systems included in the backup. The information is displayed in the following form: `Mount_Device Blocks Blocks_Free Blocks_Used Mount_Point` The `Blocks` and `Blocks_Free` fields describe the number of 512-byte blocks and the free 512-byte blocks that are present in the mounted file system. The `Blocks_Used` describes the number of 512-byte blocks used in the portion of the mounted file system mounted from the WPAR. The **-M** flag is mutually exclusive with the **-l**, **-L**, and **-N** flags.
-N	Lists information about any NFS-mounted file systems included in the backup. The information is of the following form: `RemoteHost HostFilesystem Blocks Blocks_Free Blocks_Used` The `Blocks` and `Blocks_Free` describe the number of 512-byte blocks and the free 512-byte blocks in the remote file system. The `Blocks_Used` describes the number of 512-byte blocks used in the portion of the remote file system mounted from the WPAR. The **-N** flag is mutually exclusive with the **-l**, **-L**, and **-M** flags.
-V	Verifies a tape backup. You must specify the **-f** flag with the **-V** flag. The flag is valid only for tape devices. The **-V** flag verifies the readability of the header of each file on the volume group backup and prints any errors that occur to the **stderr** file.

Examples

1. To list the contents of the workload partition backup that is located on the default device **/dev/rmt0**, use the following command:
   ```
   lssavewpar
   ```
2. To list the contents of the system backup that is located on device **/dev/cd1**, use the following command:
   ```
   lssavewpar -f /dev/cd1
   ```
3. To display colon-separated lpp information about a workload partition backup tape that is located on **/dev/rmt0**, use the following command:
   ```
   lssavewpar -Lc -f /dev/rmt0
   ```
4. To list volume group and general backup data about a backup located at `/tmp/mybackup`, use the following command:
   ```
   lssavewpar -l -f /tmp/mybackup
   ```
5. To verify the readability of each header on a workload partition backup tape in **/dev/rmt0**, use the following command:
   ```
   lssavewpar -V -f /dev/rmt0
   ```
6. To list the contents of the system backup located on device **/dev/usbms0,** use the following command:
   ```
   lssavewpar –f /dev/usbms0
   ```

Related reference:

"lslpp Command" on page 432

Related information:

savewpar 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/audit/hosts**
- **/etc/security/lastlog**
- **/etc/security/limits**
- **/etc/security/login.cfg**
- **/usr/lib/security/mkuser.default**
- **/etc/nscontrol.conf**
- **/etc/security/passwd**
- **/etc/security/portlog**
- **/etc/security/pwdalg.cfg**
- **/etc/security/roles**
- **/etc/security/smitacl.user**
- **/etc/security/smitacl.group**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/rtc/rtcd_policy.conf**

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

Item	Description
-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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.
To get the full functionality of the command, besides the **accessauths**, the role should also have the **aix.security** authorization.

On a Trusted AIX system, only users with authorization aix.mls.clear.read can list clearance attributes of other users. Only users with authorization aix.mls.tty.read can list port attributes.

Files Accessed:

Mode	File
r	**/etc/security/environ**
r	**/etc/security/group**
r	**/etc/security/audit/hosts**
r	**/etc/security/lastlog**
r	**/etc/security/limits**
r	**/etc/security/login.cfg**
r	**/usr/lib/security/mkuser.default**
r	**/etc/nscontrol.conf**
r	**/etc/security/passwd**
r	**/etc/security/portlog**
r	**/etc/security/pwdalg.cfg**
r	**/etc/security/roles**
r	**/etc/security/smitacl.user**
r	**/etc/security/smitacl.group**
r	**/etc/security/user**
r	**/etc/security/user.roles**
r	**/etc/security/domains**
rw	**/etc/security/rtc/rtcd_policy.conf**

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

Item	Description
/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/audit/hosts	Contains host and processor IDs.
/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.
/etc/nscontrol.conf	Contains configuration information of some name services.
/etc/security/passwd	Contains password information.
/etc/security/portlog	Contains unsuccessful login attempt information for each port.
/etc/security/pwdalg.cfg	Contains configuration information for loadable password algorithms (LPA).
/etc/security/roles	Contains a list of valid roles.
/etc/security/smitacl.user	Contains user ACL definitions.
/etc/security/smitacl.group	Contains group ACL definitions.
/etc/security/user	Contains the extended attributes of users.
/etc/security/user.roles	Contains a list of roles for each user.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.
/etc/security/domains	Contains the valid domain definitions for the system.
/etc/security/rtc/rtcd_policy.conf	Contains configuration information for the **rtcd** daemon

Related information:

usrck command

getportattr command

List of Security and Auditing Subroutines

Trusted AIX

lssecattr Command

Purpose

Displays the security attributes of a command, a device, a privileged file, a process or, a domain-assigned object.

Syntax

lssecattr [**-R load_module**] { **-c** | **-d** | **-p** [**-h**] [**-A**] | **-f** | **-o** } [**-C** | **-F**] [**-a** *List*] { **ALL** | *Name* [,*Name*] ... }

Description

The **lssecattr** command lists the security attributes of one or more commands, devices, or processes. The command interprets the *Name* parameter as either a command, a device, a privileged file, a process, or a domain-assigned object based on whether the **-c** (command), **-d** (device), **-f** (privileged file), **-p** (process), or **-o** (domain-assigned object) flag is specified. If the **-c** flag is specified, the *Name* parameter must

include the full path to the commands. If the **-d** flag is specified, the *Name* parameter must include the full path to the devices. If the **-f** flag is specified, the *Name* parameter must include the full path to the file. If the **-p** flag is specified, the *Name* parameter must be the numeric process identifier (PID) of an active process on the system. If the **-o** flag is specified, the *Name* parameter must be the full path if it is a file or device and for port or port ranges it must be prefixed with TCP_ or UDP_. Use the **ALL** keyword to list the security attributes for all commands, devices, files, or processes. By default, the **lssecattr** command displays all of the security attributes for the specified object. To view the selected attributes, use the **-a** *List* flag.

If the system is configured to use databases from multiple domains, the privileged commands, privileged devices, and privileged files, as specified by the *Name* parameter, are searched from the domains in the order specified by the **secorder** attribute of the corresponding database stanza in the **/etc/nscontrol.conf** file. If duplicate entries exist in multiple domains, only the first entry instance is listed. Use the **-R** flag to list the objects from a specific domain.

By default, the **lssecattr** command lists the security attributes on one line. It displays the attribute information as the definitions of `Attribute=Value`, each separated by a blank space. To list the attributes in stanza format, use the **-F** flag. To list the attributes as colon-separated records, use the **-C** flag.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable requires a blank space between attributes to list multiple attributes. If you specify an empty list, only the object names are displayed. The attributes that can be listed in the *List* variable are dependent on which one of the **-c**, **-d**, and **-p** flags is specified. For a list of the valid attribute names for each flag, see the **setsecattr** command.
-A	Display the list of authorizations used by a specified process. This flag can only be used with the **-p** flag.
-c	The *Name* parameter specifies the full paths to one or more commands on the system that have entries in the **/etc/security/privcmds** privileged command database.

-C	Displays the privileged security attributes in colon-separated records as follows:
	`#name:attribute1:attribute2: ...`
	`name:value1:value2: ...`
	`name:value1:value2: ...`
	
	The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If the **-a** flag is specified, the order of the attributes matches the order specified in the **-a** flag. If an object does not have a value for a given attribute, the field is still output but is empty. The last field in each entry is terminated by a newline character rather than a colon.

-d	The *Name* parameter specifies the full paths to one or more devices on the system that have entries in the **/etc/security/privdevs** privileged device database.
-f	The *Name* parameter specifies the full paths to one or more files on the system that have entries in the **/etc/security/privfiles** privileged files database.

-F	Displays the output in stanza format, with each stanza identified by an object name. Each pair of `Attribute=Value` is listed on a separate line:
	`Name:`
	`attribute1=value`
	`attribute2=value`
	`attribute3=value`

-h	Displays the full hierarchy of privileges for the process. By default, only the highest level of privilege is listed.
-o	The *Name* parameter specifies one of the following entries in the **/etc/security/domobjs** domain-assigned object database. • the full paths to one or more devices/files on the system • the port or port ranges prefixed with TCP_ or UDP_ • the network interfaces

Item	Description
-p	The *Name* parameter specifies the numeric process identifiers (PID) of one or more active processes on the system. The **-p** flag cannot be listed with the **-R** flag as they are mutually exclusive.
-R *load_module*	Specifies the loadable module to query the *Name* entry from.

Parameters

Item	Description
ALL	For all commands, devices or processes.
Name	The object to modify. The *Name* parameter is interpreted according to which one of the **-c**, **-d**, **-p**, and **-o** flags is specified.

Security

The **lssecattr** command is a privileged command. It is owned by the root user and the security group, with mode set to 755. You must assume a role with at least one of the following authorizations to run the command successfully.

Item	Description
aix.security.cmd.list	Required to list the attributes of a command with the **-c** flag.
aix.security.device.list	Required to list the attributes of a device with the **-d** flag.
aix.security.file.list	Required to list the attributes of a file with the **-f** flag.
aix.security.proc.list	Required to list the attributes of a process with the **-p** flag.
aix.security.dobject.list	Required to list the attributes of a domain-assigned object with the **-o** flag.

File Accessed

Item	Description
File	**Mode**
/etc/security/privcmds	r
/etc/security/privdevs	r
/etc/security/privfiles	r
/etc/security/domobjs	r

Examples

1. To display the access authorization and the innate privileges of the **/usr/sbin/mount** command, enter the following command:

   ```
   lssecattr -c -a accessauths innateprivs /usr/sbin/mount
   ```

2. To display all the security attributes of the **/dev/mydev** device, enter the following command:

   ```
   lssecattr -d /dev/mydev
   ```

3. To display all the security attributes of the **/dev/mydev** device in LDAP, enter the following command:

   ```
   lssecattr -R LDAP -d /dev/mydev
   ```

4. To display the privileges for the effective and used privilege sets of two processes in a colon format, enter the following command:

   ```
   lssecattr -p -C -a eprivs uprivs 38483,57382
   ```

5. To display the read authorization list of the **/etc/security/user** file, enter the following command:

   ```
   lssecattr -f -a readauths /etc/security/user
   ```

6. To display the used authorizations for a process in a stanza format, enter the following command:

   ```
   lssecattr -F -p -A 34890
   ```

7. To display all the domain attributes of the **/dev/dev1** device, enter the following command:

```
lssecattr -o /dev/dev1
```

8. To display all the domain attributes of the network interface **en0** device, enter the following command:

```
lssecattr -o en0
```

Related information:

setsecattr command

pvi command

getcmdattr subroutine

/etc/security/privdevs file

---

lssecmode Command

Purpose

Displays the current or pending security mode configuration and key types in a formatted output.

Syntax

lssecmode [**-p**] [**-d** | **-D** *delim*] [**-x**] [**-T**] [**-V**] [**-h**]

Description

The **lssecmode** command displays information about the current or pending security mode configuration and key types. This information consists of the compliance mode, public or private key type, and default symmetric key types.

Note: If no flag is specified, the current security configuration mode and key types are displayed.

Flags

Item	Description
-d	Displays the delimiter-formatted output. The default delimiter is a colon (:). You can use the **-D** flag to change the default delimiter.
-D *delim*	Specifies the delimiter to be used in the formatted output. By default, the colon (:) character is used as the delimiter in the output. You can use this flag to format the output with another delimiter that can contain one or more characters. For example, if the output data already contains colons, the default delimiter might result in confusion.
-h	Displays the command usage.
-p	Displays the pending security mode configuration and key types.
-T	Writes the command trace messages to standard output.
-V	Writes the command verbose messages to standard output.
-x	Specifies that the header information must not be displayed.

Exit status

0 Successful.

1 Missing argument error.

2 Invalid option error.

3 API error.

Examples

1. To display the current security configuration mode and key types, enter the following command:

   ```
   # lssecmode
   ```

   An output similar to the following example is displayed:

   ```
   Current Security Mode Configuration
   Compliance Mode : none
   Asymmetric Key Type : rsa512
   Symmetric Key Type : default
   ```

2. To display the pending security configuration mode and key types, enter the following command:

   ```
   # lssecmode -p
   ```

   If there is no pending security configuration mode and if the staging file is not present, the following output is displayed:

   ```
   2650-384 There are no pending configuration available
   ```

3. To format the output with a delimiter **::**, enter the following command:

   ```
   # lssecmode -D "::"
   ```

   An output similar to the following example is displayed:

   ```
   Current Security Mode Configuration
   Compliance Mode :: none
   Asymmetric Key Type :: rsa512
   Symmetric Key Type :: default
   ```

4. To display the pending security configuration mode and key types with command verbose messages and without header information, enter the following command:

   ```
   # lssecmode -p -x -V -D "::" -T
   ```

   An output similar to the following example is displayed:

   ```
   Invoked with parameters: -p -x -V -D :: -T
   Invoking lssecmode to get the pending security mode and key types....
   No header information required..
   Compliance Mode :: none
   Asymmetric Key Type :: rsa512
   Symmetric Key Type :: default
   Checking lssecmode log file size and backup if necessary....
   No log file exist. No backup is needed
   ```

Location

/opt/rsct/bin/lssecmode
Contains the **lssecmode** command.

---

lssensor Command

Purpose

Displays information about sensors and microsensors that are defined to the resource monitoring and control (RMC) subsystem.

Syntax

lssensor [**-m**] [**-a** | **-n** *host1*[**,***host2*...] | **-N** { *node_file* "–" }] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-v** | **-V**] [**-A** | *sensor_name1* [*sensor_name2*...]]

Description

The **lssensor** command displays the attributes of one or more sensors. If you do not specify any *name* parameters, the **lssensor** command lists the names of all of the sensors. Use the **-A** flag to list all of the sensors and all of their attributes and values. Use the **-m** flag to display information about microsensors.

The **lssensor** command displays values for attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. A sensor command is a command or script that the sensor resource manager runs to set and update a sensor's attribute values. A microsensor module is a loadable module that the microsensor resource manager runs to set and update a microsensor's attribute values.

Use the **-l**, **-t**, **-d**, or **-D** flags to display the output in long format, table format, or delimiter format. The **-x** flag omits headings when any of these flags are used.

The **lssensor** command runs on any node. If you want **lssensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **lssensor** to run on a subset of nodes in a domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

The **lssensor** command lists the following information about defined sensors:

Field	Description
Name	The name of the sensor.
Command	The command that is run to update the sensor attributes
ConfigChanged	Information about changes to access or to persistent attributes
ControlFlags	Indicates whether any special handling is required for this sensor
Description	This field is not used
ErrorExitValue	Indicates how the exit value is interpreted by the sensor resource manager
ErrorMessage	This field is not used
ExitValue	The exit code from the command that is running
Float32	The type **float32** attribute for this sensor resource
Float64	The type **float64** attribute for this sensor resource
Int32	The type **int32** attribute for this sensor resource
Int64	The type **int64** attribute for this sensor resource
MonitorStatus	This attribute is set to 1 when certain sensor attributes are being monitored
NodeNameList	The name of the node where the sensor resource is defined
RefreshInterval	The interval (in seconds) during which the sensor attribute values are updated when the sensor command is run
SavedData	An output string from the sensor command
SD	Contains all dynamic resource attributes except **ConfigChanged**, **Quantum**, and **ExitValue** as its elements
String	The type string attribute for this sensor resource
TimeCommandRun	Indicates the date and time that the sensor command was run
Uint32	The type **uint32** attribute for this sensor resource
Uint64	The type **uint64** attribute for this sensor resource
UserName	The user ID that is used when run the sensor command is run

The **lssensor** command displays the following information about defined microsensors:

Field	Description
Name	The name of the microsensor.
ActivePeerDomain	The peer domain for which information is being displayed.
Arguments	The arguments for this microsensor resource.
ConfigChanged	Information about changes to persistent attributes or to access.
CustomDynamicAttributes	The custom dynamic attributes for this microsensor resource.
Description	Information about the microsensor and what it monitors.
Float32	The type float32 attribute for this microsensor resource.
Float32Array	The type float32 array attribute for this microsensor resource.
Float64	The type float64 attribute for this microsensor resource.
Float64Array	The type float64 array attribute for this microsensor resource.
Int32	The type int32 attribute for this microsensor resource.
Int32Array	The type int32 array attribute for this microsensor resource.
Int64	The type int64 attribute for this microsensor resource.
Int64Array	The type int64 array attribute for this microsensor resource.
LastQueryRC	The return code from the microsensor module from the last time the microsensor was called for an attribute of the microsensor resource.
LastQueryTime	The time of LastQueryRC.
ModuleName	The path name to the loadable microsensor module.
MonitorStatus	This attribute is set to 1 when any of the other microsensor attributes is being monitored.
NodeNameList	The name of the node where this microsensor is defined.
RefreshInterval	The interval (in seconds) during which the microsensor attribute values are updated when the microsensor callback is called.
String	The type string attribute for this microsensor resource.
StringArray	The type string array attribute for this microsensor resource.
UInt32	The type uint32 attribute for this microsensor resource.
UInt32Array	The type uint32 array attribute for this microsensor resource.
UInt64	The type uint64 attribute for this microsensor resource.
UInt64Array	The type uint64 array attribute for this microsensor resource.

Flags

-a Lists sensors that match the specified name on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE 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 domain and a peer domain exist, **lssensor** -a with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A Displays all of the sensors with their attributes and values.

–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 that the information be displayed in "long" format. Each attribute is displayed on a separate line.

-m Specifies that information about microsensors will be displayed.

-n *host1*[**,***host2***...**]
Specifies the node from which the sensor should be listed. By default, the sensor is listed from the local node. This flag is only appropriate in a management domain or a peer domain.

-N {*node_file* | **"-"**}
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "–"** in a management domain or a peer domain to read the node names from standard input.

–t Specifies table format. Each attribute is displayed in a separate column, with one sensor resource per line.

-x Suppresses header printing when **-l**, **-t**, **-d**, or **-D** is specified.

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

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

Parameters

sensor_name1 [*sensor_name2***...**]
Specifies the names of one or more sensors to display.

Security

To display sensor information using this command, you need read permission for the **IBM.Sensor** resource class. To display microsensor information using this command, you need read permission for the **IBM.MicroSensor** resource class. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* 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.

6 No sensor resources were found.

n Based on other errors that can be returned by the RMC subsystem.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Examples

1. To list the names of all of the sensors, enter:

```
lssensor
```

The output will look like this:

```
sensor1
sensor2
sensor3
```

2. To list the names and attributes of all sensors, enter:

```
lssensor -A
```

The output will look like this:

```
 Name = sensor1
 ActivePeerDomain =
 Command = /usr/local/bin/sensorcmd1
 ConfigChanged = 0
 ControlFlags = 1
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 1.06381e+06
 Float64 = 1.06381e+06
 Int32 = 1063814
 Int64 = 1063814
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData = Last SavedData
 SD = [string from sensor1,1063814,1063814,1063814,1063814,1.06381e+06,1.06381e+06]
 String = string from sensor1
 Uint32 = 1063814
 Uint64 = 1063814
 UserName = root
-------------------
 Name = CFMRootModTime
```

```
 ActivePeerDomain =
 Command = /opt/csm/csmbin/mtime/cfmroot
 ConfigChanged = 0
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
------------------
 Name = ErrorLogSensor
 ActivePeerDomain =
 Command = /opt/csm/csmbin/monerrorlog
 ConfigChanged = 0
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
------------------
.
.
.
```

3. To list the attributes of **sensor2**, enter:

```
lssensor sensor2
```

The output will look like this:

```
Name = sensor2
Command = /usr/local/bin/sensorcmd2
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ExitValue = 127
Float32 = 0
Float64 = 0
Int32 = 0
Int64 = 0
NodeNameList = {somenode.pok.ibm.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
```

```
String =
Uint32 = 0
Uint64 = 0
UserName = root
```

4. To list all of the sensors' information using delimited output, enter:

   ```
   lssensor -dA
   ```

   The output will look like this:

   ```
   Displaying sensor information:
   Name:ActivePeerDomain:Command:ConfigChanged:ControlFlags:Description:ErrorExitValue:ErrorMessage:ExitValue:
   Float32:Float64:Int32:Int64:MonitorStatus:NodeNameList:RefreshInterval:SD:SavedData:
   String:TimeCommandRun:Uint32:Uint64:UserName:
   JoeExample:JoeDomain:cat /etc/motd:0:0::1::0:
   ::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
   :Fri Feb  6 19:00:00 2009:::root:
   JoeSample:JoeDomain:/opt/rsct/install/bin/ctversion:0:0::1::0:
   ::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
   :Fri Feb  6 19:00:00 2009:::root:
   JoeSens:JoeDomain:/tmp/sensor/numusers:0:1::1::0:
   ::::0:{node1.myhost.com}:0:[,2,0,0,0,0,0]::
   :Tue Mar  3 10:27:19 2009:::root:
   ```

5. To list the names of all of the sensors on the nodes that are listed in the **/u/joe/common_nodes** file, enter:

   ```
   lssensor -N /u/joe/common_nodes
   ```

   where /u/joe/common_nodes contains:

   ```
   # common node file
   #
   node1.myhost.com    main node
   node2.myhost.com    backup node
   ```

   The output will look like this:

   ```
   sensor1
   sensor2
   sensor3
   ```

6. To list the names of all of the microsensors, enter:

   ```
   lssensor -m
   ```

   The output will look like this:

   ```
   IBM.MSensor1
   IBM.MSensor2
   IBM.MSensor3
   ```

7. To list the attributes of the microsensor **IBM.MSensor2**, enter:

   ```
   lssensor -m IBM.MSensor2
   ```

   The output will look like this:

   ```
   Name = IBM.MSensor2
   ActivePeerDomain =
   Arguments = all
   ConfigChanged = 0
   CustomDynamicAttributes = {[CDA1,19,1,3,0,1],[CDA2,20,2,2,0,1],[CDA3,21,3,2,0,1]
   Description =
   Float32 =
   Float32Array =
   Float64 =
   Float64Array =
   Int32 = 52
   Int32Array = {36, 45, 2, 73}
   Int64 =
   Int64Array =
   LastQueryRC = 0
   LastQueryTime = Tue Mar 31 18:00:00 2009
   ModuleName = /usr/slib/msensors/sensor2
   ```

```
MonitorStatus = 0
NodeNameList = {node2.gumby.com}
RefreshInterval = 600
String =
StringArray =
UInt32 =
UInt32Array =
UInt64 =
UInt64Array =
```

Location

/opt/rsct/bin/lssensor

---

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 **lsslot** command is used to list the connectors which are connection points for either physical entities like PCI adapters or logical entities like logical slots or logical host-Ethernet adapter ports. The command can list the following types of connectors:

- pci: a physical connector
- slot: a logical connector
- phb: a logical connector
- port: a logical connector

The **-a** and the **-o** flags will be ignored for the logical connectors. The **lsslot** command in the case of the logical connectors displays the logical entities that are currently assigned to the partition, depending upon the connector type specified. When there are multiple slots under a PHB, a logical slot entity can be associated with a logical slot connector. Otherwise, it can be associated with a logical PHB connector. You can run the **lsslot -c** *slot* command and the **lsslot -c** *phb* command to view all logical slot entities.

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

When the PHBs are listed using the **lsslot** command, the Device(s) Connected column will display the ODM name of the PHB followed by the ODM names of the devices corresponding to the logical slots underneath the PHB, with all the ODM devices associated with each logical slot displayed on each

separate line under the ODM name of the PHB. In case there is no ODM name for the PHB, a blank line will be displayed.

Flags

Item	Description
-a	Displays available hot plug slots and their characteristics. Available slots are those slots that do not have a hot plug device connected. This flag is ignored for connector types of slot and phb.
-c *ConnectorType*	Displays the slots of the specified *ConnectorType*. *ConnectorType* identifies the type of connector. For example, the *ConnectorType* for a hot plug PCI slot is pci, for logical slots, it is slot and for PHBs, it is phb. 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 associated. 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. This flag is ignored for connector types of slot and phb.
-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:

Item	Description	
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:

Item	Description	
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:

Item	Description	
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.

4. To list all the PCI Host Bridges that are assigned to the partition, enter:

   ```
   lsslot -c phb
   ```

This displays output similar to the following:

```
PHB Name         Description                 Device(s) Connected
PHB 1            Logical PCI Host Bridge     pci0
                                             pci2 scsi1
PHB 2            Logical PCI Host Bridge     pci1
                                             pci3
                                             pci4 scsi2
```

5. In case the PCI Host Bridge is assigned to the partition but has no ODM data, column will show blank as shown in this example. For example, when you enter:

   ```
   lsslot -c phb
   ```

   The output will look similar to the following:

   ```
   PHB Name         Description              Device(s) Connected
   PHB 4            Logical PCI Host Bridge
   PHB 5            Logical PCI Host Bridge
   ```

6. To list all the logical host-Ethernet adapter-port devices that are assigned to the partition, enter:

   ```
   lsslot  -c port
   ```

Item	Description	
LHEA port name	**Description**	**Device(s) Connected**
Port 1	Logical HEA Port	ent4
Port 2	Logical HEA Port	ent7

When the logical host-Ethernet adapter port is assigned to the partition but has no ODM data, the column shows `Unknown` as shown in the example:

```
lsslot  -c port
```

Item	Description	
LHEA port name	**Description**	**Device(s) Connected**
Port 4	Logical HEA Port	Unknown
Port 5	Logical HEA Port	Unknown

Files

Item	Description
/usr/sbin/lsslot	Contains the **lsslot** command.

Related reference:

"lsdev Command" on page 396

Related information:

drslot command

PCI hot plug management

lssrad Command

Purpose

Displays the system SRADID (Scheduler Resource Allocation Domain Identifier) hierarchy and topology.

Syntax

lssrad [-v] { **-s** *SRADID* | **-a** }

Description

The **lssrad** command displays information related to SRADIDs, such as the processor and memory associated with the SRAD (Scheduler Resource Allocation Domain) and REF1 system detail level, where REF1 is the first hardware provided reference point that identifies sets of resources that are near each other. This command also displays the SRADID hierarchy and topology.

Flags

Item	Description
-a	Displays all SRADs in the system.
-s *SRADID*	Displays the specified SRADID.
-v	Displays resources in the SRAD, along with the REF1 System Detail Level that the SRAD belongs to, in verbose mode.

Examples

1. To display the list of all SRADs in the system, enter:

   ```
   # lssrad -a
   ```

2. To verify that a specific SRAD exists, enter:

   ```
   # lssrad -s 0
   SRAD
   0
   ```

   ```
   # lssrad -s 5
   SRAD 5:  No such SRAD
   ```

3. To display the topology of a specific SRAD, enter:

   ```
   # lssrad -v -s 5
   ```

4. To display the SRADID hierarchy and topology, enter:

   ```
   # lssrad -v -a
   ```

Files

Item	Description
/usr/sbin/lssrad	Contains the **lssrad** command

---

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.

The **lssrc** command output can sometimes show two entries for a particular daemon. One instance will be active and another instance will be inoperative. This can happen if the subsystem is modified (using the **mkssys** command or **chssys** command) without stopping the subsystem. The original subsystem will remain active and the modified instance will be inoperative until the subsystem is stopped and started again.

Flags

Item	Description
-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.

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Exit status

0 The command ran successfully.

1 The command failed.

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

Item	Description
/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 reference:

"mkssys Command" on page 825

Related information:

rmssys command

System Resource Controller

---

lsts Command

Purpose

Lists thin server information. This command can be run on a thin server.

Syntax

lsts [[**-l**{**1**|**2**|**3**}] . . .] [**-v**] [*ThinServer*]

Description

The **lsts** command lists information pertaining to a thin server. The level of information to be listed depends on the numeric value specified by the **-l** flag, with a level ranging from 1 - 3 (3 being the most detailed). If a level is not specified, a default of level 1 information is displayed. This command can be run on both a NIM master or a thin server. When run on a NIM master and no argument is provided, the **lsts** command lists all thin servers in the environment controlled by the caller of the **lsts** command.

Flags

Item	Description
-l{**1**|**2**|**3**}	Specifies the level of output. **1** This level displays very limited information related to a thin server. The information listed shows only a brief summary of the thin server, such as the common image it is using. **2** This level displays more than just basic information related to a thin server. The level includes information pertaining to the software content of the thin server. **3** This level displays more in-depth information related to a thin server. The level includes information pertaining to the installation log of the thin server.
-v	Enables verbose debug output when the **lsts** command runs.

Parameters

Item	Description
Thinserver	Specifies the thin server where the command lists information about the client.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **lsts** command.

Examples

1. To list brief status information for a thin server named `lobo`, enter:

   ```
   lsts lobo
   ```

   Information similar to the following is displayed:

   ```
   Lobo:
     class          = machines
     type           = diskless
     platform       = chrp
     netboot_kernel = 64
     if1            = master_net jsblade04 0 ent1
     cable_type1    = bnc
     Cstate         = diskless or dataless boot is enabled
     prev_state     = in the process of booting
     Mstate         = currently running
     boot           = boot
     dump           = dump_res
     paging         = paging_res
     root           = root_res
     spot           = 530spot_res
     cpuid          = 00012A80D000
     control        = master
     Cstate_result  = success
   ```

2. To list software content for a thin server named `lobo`, enter:

   ```
   lsts -l2 lobo
   ```

   Software content similar to the following is displayed from the common image:

   ```
   Fileset                  Level  State  Type  Description
   (Uninstaller)
   ----------------------------------------------------------
   bos.64bit               5.2.0.75  C     F    Base Operating System 64 bit Runtime
   bos.diag.com            5.2.0.75  C     F    Common Hardware Diagnostics
   bos.diag.rte            5.2.0.75  C     F    Hardware Diagnostics
   .
   .
   .
   ```

3. To list both software content and status information for a thin server named `lobo`, enter:

   ```
   lsts -l1 -l2 lobo
   ```

Location

/usr/sbin/lsts

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

Related information:

dbts command

nim command

nimconfig command

swts command

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"imptun Command" on page 35

"mktun Command" on page 839

Related information:

gentun command

lstxattr Command

Purpose

Lists the security attributes of a file, directory, process, or interprocess communication (IPC).

Syntax

lstxattr { **-f** | **-m** | **-p** | **-q** | **-s** } [**-C** | **-F**] [**-a** *List*] { *Name* [*,Name*] ... }

Description

The **lstxattr** command lists Trusted AIX security attributes of the file, process, shared memory, message queue or semaphore that is specified by the *Name* parameter. The command interprets the *Name* parameter as either a file, process, shared memory, message queue or semaphore based on whether the **-f** (file), **-p** (process), **-m** (shared memory), **-q** (message queue) or **-s** (semaphore) flag is specified.

By default, the **lstxattr** command displays all the security attributes for the specified object. To view the selected attributes, use the **-a** *List* flag.

By default, the security attributes are listed on one line. The command displays the attribute information as *Attribute* = *Value* definitions, each separated by a blank space. To list the attributes in stanza format, use the **-F** flag. To list the attributes as colon-separated records, use the **-C** flag.

Flags

Item | **Description**

-a *List*

Lists the attributes to display. The *List* variable requires a blank space between attributes to list multiple attributes. If you specify an empty list, the command displays only the object names. The attributes that can be listed in the *List* variable are dependent on which one of the **-f**, **-p**, **-m**, **-q** or **-s** flags that you specified.

Use the following file security attributes for the **-f** flag:

sl Sensitivity Label. If specified for a non-regular file, the command lists both the maximum and minimum sensitivity labels.

maxsl Maximum Sensitivity Label. If specified for regular files, the command lists the **sl** value.

minsl Minimum Sensitivity Label. If specified for regular files, the command lists the **sl** value.

tl Integrity Label.

secflags Trusted AIX file security flags.

Use the following process security attributes for the **-p** flag:

effsl Effective Sensitivity Label.

maxcl Maximum Sensitivity Clearance Label.

mincl Minimum Sensitivity Clearance Label.

efftl Effective Integrity Label.

maxtl Maximum Integrity Label.

mintl Minimum Integrity Label.

Use the following security attributes for the **-q, -m,** and **-s** flags:

sl Sensitivity Label.

tl Integrity Label.

Item	Description

-C	Displays the privileged security attributes in colon-separated records in the following way:
	`#name:attribute1:attribute2: ...`
	`name:value1:value2: ...`
	`name:value1:value2: ...`
	
	The output is preceded by a comment line that lists details about the attribute represented in each colon-separated field. If you specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If an object does not have a value for a given attribute, the field is still displayed but is empty. The last field in each entry is ended by a newline character rather than a colon.

-f	Lists the security attributes of a file. The *Name* parameter specifies the path to this file on the system.

-F	Displays the output in stanza format, with each stanza identified by a object name. Each *Attribute* = *Value* pair is listed on a separate line:
	`Name:`
	`attribute1=value`
	`attribute2=value`
	`attribute3=value`

-m	Lists the security attributes of a shared memory. The *Name* parameter specifies the numeric shared memory identifier on the system.
-p	Lists the security attributes of a process. The *Name* parameter specifies the numeric process identifier (PID) of an active process on the system.
-q	Lists the security attributes of a message queue. The *Name* parameter specifies the numeric message queue identifier on the system.
-s	Lists the security attributes of a semaphore. The *Name* parameter specifies the numeric semaphore identifier on the system.

Parameters

Item	Description
Name	The object to list. The *Name* parameter is interpreted according to which one of the **-f**, **-p**, **-m**, **-q** or **-s** flags that you specified.

Security

The **lstxattr** command is a privileged command. It is owned by the root user and the security group, with the mode set to 755.

Restriction: The binary labels of the objects are interpreted as human-readable format and depend on the values in the **/etc/security/enc/LabelEncodings** file. If the conversion fails, you must have the following authorizations:

- **aix.mls.stat** authorizations for listing the binary labels of files and IPC objects
- **aix.mls.proc** authorizations for listing the binary labels of processes

Files Accessed:

Item	Description
Mode	**File**
r	**/etc/security/enc/LabelEncodings**

Examples

1. To list all the attributes of the **regfile** file, enter the following command:
   ```
   lstxattr –f regfile
   ```
2. To list the maximum sensitivity, minimum sensitivity and integrity labels of the **dirname** directory, enter the following command:
   ```
   lstxattr –f -a maxsl minsl tl dirname
   ```

3. To list the labels of a message-queue IPC object with "0" as the message queue ID, enter the following command:

   ```
   lstxattr –q –a sl tl 0
   ```

4. To list the labels of a shared-memory IPC object with "3145728" as the shared memory ID, enter the following command:

   ```
   lstxattr –m –a sl tl 3145728
   ```

Related reference:

"ipcs Command" on page 122

Related information:

settxattr command

Trusted AIX

---

lsuser Command

Purpose

Displays user account attributes.

Syntax

lsuser [**-R** *load_module*] [**-c** | **-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, but does not display empty attributes.

Note: If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file, the **lsuser** command lists the merged group from the LDAP module and the LOCAL module, if present.

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** or **-C** flag.

You can use the System Management Interface Tool (SMIT) **smit lsusers** fast path to run this command.

Flags

Item	Description
-a *List*	Lists the attributes to display. The *List* variable can include any attribute that is 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: ...`
	If a value contains a **:** symbol, then in the output **:** symbol is prefixed with the **#!** symbols.

Item	Description

-C	Displays the user attributes in colon-separated records that are easier to parse than the output of the **-c** flag:
	`#name:attribute1:attribute2: ...`
	`User1:value1:value2: ...`
	`User2:value1:value2: ...`
	
	The output is preceded by a comment line that has details about the attribute represented in each colon-separated field. If you also specify the **-a** flag, the order of the attributes matches the order specified in the **-a** flag. If you do not have a value for a given attribute, the field is still displayed, but is empty. If a value contains a **:** symbol, then in the output the **:** symbol is prefixed with **#!** symbols.The last field in each entry ends with a newline character rather than a colon.

-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 that is used to display the user account attributes. If the *domainlessgroups* attribute is set in the /etc/secvars.cfg file and the **-R** *LDAP* command is used, the attribute list is obtained from the LOCAL module. This condition applies if the user exists on the LOCAL module, and does not exist on the LDAP module. This condition also applies to the **-R** *files* command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details to the type of failure.

Security

Access Control: This command must 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 starts the command, some users might not be able to access all the information. This command must have the *trusted computing base* attribute.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.
To get the full functionality of the command, besides the **accessauths**, the role must also have the **aix.security.user.audit** authorization.

On a Trusted AIX system, only users with authorization aix.mls.clear.read can list clearance attributes of other users. See Trusted AIX in the *Security* for more information.

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
r	/etc/security/enc/LabelEncodings

Examples

1. To display the user `id` and group-related information about the `smith` account in stanza form, enter the following command:

   ```
   lsuser -f -a id pgrp groups admgroups smith
   ```

   Information similar to the following is displayed:

   ```
   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 the following command:

   ```
   lsuser -c -a id home groups smith
   ```

   Information similar to the following is displayed:

   ```
   # 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 the following command:

   ```
   lsuser smith
   ```

   All the attribute information is displayed, with each attribute separated by a blank space.

4. To display all the attributes of all the users, enter the following command:

   ```
   lsuser ALL
   ```

   All the attribute information is displayed, with each attribute separated by a blank space.

Files

Item	Description
/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.
/etc/security/enc/LabelEncodings	Contains label definitions for the Trusted AIX system.

Related reference:

"mkgroup Command" on page 719

Related information:

chsh command

passwd command
setgroups command

lsusil Command

Purpose

Lists one or more user-specified installation location (USIL) instances.

Syntax

lsusil [**-R** *RelocatePath* | **ALL**]

Description

The **lsusil** command lists one or more USIL instances.

Flags

Item	Description
-R *RelocatePath*	The path to an existing USIL location

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/lsusil	Contains the **lsusil** command.

Related reference:
"mkusil Command" on page 844
Related information:
chusil command
rmusil command

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

Item	Description
-a	Lists all stanzas in the **/etc/vfs** file, including the default stanza.

Parameter

Item	Description
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

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.

Related reference:
"mount Command" on page 885
Related information:
chvfs command
File systems
/etc/vfs file

---

lsvg Command

Purpose

Displays information about volume groups.

Syntax

lsvg [**-L**] [**-o**] | [**-n** *descriptorphysicalvolume*] | [**-i**] [**-l** | **-M** | **-p**] [**-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 System Management Interface Tool (SMIT) **smit lsvg** fast path to run this command.

Flags

Item	Description
-i	Reads volume group names from standard input.
-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.
-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 data.
-m	Lists the mirror pool that each logical volume copy in the volume group belongs to.
-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.

Item	Description
-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.
-P	Lists the mirror pool that each physical volume in the volume group belongs to.

Information displayed if you do not specify any flags:

Item	Description
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).
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. This information is displayed only for 32 and 128 PV volume groups.
LTG size	Logical track group size of the volume group. The maximum amount of data that can be transferred in one I/O request to the disks of the volume group. The LTG size will be displayed in kilobytes unless the LTG size is greater than 1 MB, in which case megabytes will be used. If the volume group was created on AIX 5.3, then it is capable of dynamically determining the LTG size based-on the disk topology and it is listed as Dynamic. If that capability is disabled by the user with the **varyonvg -M** option, then it will be listed as Static. If the capability does not exist because the volume group was created prior to AIX 5.3, then the VG will not be listed as Static or Dynamic.
BB POLICY	Bad block relocation policy of the volume group.

Item	Description
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.
PV Restriction	Displays the existing PV type restriction on the physical volumes comprising the volume group. A value of *none* indicates no PV restriction exists for the volume group. A value of *SSD* indicates the volume group has a PV restriction requiring all PVs to be *SSD* type PVs. No other values are supported.
Infinite Retry	Lists the infinite retry option of the volume group.
Critical VG	Lists whether the Critical VG option is turned on or off for the volume group.
FS SYNC OPTION	Lists whether the logical volume manager resynchronizes the blocks allocated only by the Enhanced Journaled File System (JFS2), if JFS2 is mounted.
Critical PVs	Lists Critical PVs option of the volume group. This option is available in IBM AIX 7.2 with Technology Level 1, or later.

Examples

1. To display the names of all active volume groups, enter the following command:

   ```
   lsvg  -o
   ```

2. To display the names of all volume groups within the system, enter the following command:

   ```
   lsvg
   ```

3. To display information about volume group vg02, enter the following command:

   ```
   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 the following command:

   ```
   lsvg  -l vg02
   ```

Files

Item	Description
/usr/sbin	Contains the directory where the **lsvg** command resides.

Related reference:

"lspv Command" on page 482

Related information:

chvg command

varyonvg command

Logical volume storage

lsvgfs Command

Purpose

Displays a list of file systems belonging to a volume group.

Syntax

lsvgfs *volumegroup*

Description

The **lsvgfs** command displays a list of file systems that belong to the specified volume group.

Parameters

Item	Description
volumegroup	Specifies a volume group.

Examples

1. To display a list of file systems in the volume group vg02, enter the following command:
   ```
   lsvgfs vg02
   ```

Exit Status

The **lsvgfs** command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Files

Item	Description
/usr/sbin	Contains the directory where the **lsvgfs** command resides.

Related reference:

"lsvg Command" on page 554

"lslv Command" on page 452

"lsfs Command" on page 411

Related information:

Logical volume storage

---

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 System Management Interface Tool (SMIT) **smit lsvirprt** fast path to run this command.

Flags

Item	Description
-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: **Note:** [*.*] is not a required element for the following format values. **%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

Item	Description
/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 reference:
"mkvirprt Command" on page 851
Related information:
chvirprt command
qconfig command
Configuring a printer without adding a queue

---

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:

Item	Description
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

Item	Description
/usr/bin/X11/lsvmode	Contains the **lsvnode** command.

Related information:

chvmode command

---

lsvpd Command

Purpose

Lists the vital product data (VPD) associated with the field replaceable units (FRUs) configured on a system.

Syntax

lsvpd [**-m**] [**-s** *serial_number*] [**-t** *type_model*] [**-v**]

Description

The **lsvpd** command collects vital product data (VPD) for field replaceable units (FRUs). It reads the appropriate device configuration object classes in the Object Data Manager (ODM) and gathers VPD and general system information. The **lsvpd** command can extract additional VPD by reading data structures that are specific to the platform on which it is running. Data is provided in a format that aids service personnel in monitoring device quality and performance.

Note: Output from the **lsvpd** command is informational only and subject to change as hardware definitions change. Portable applications should not parse this data.

Flags

Item	Description
-m	Distinguishes between a FRU with global VPD and a FRU with partition private VPD. FRUs with global VPD begin with a line in the format of `*FC ********`. FRUs with partition private VPD begin with a line in the format of `*FC ========`. If this flag is not specified, the output begins with a line in the format of `*FC ????????`. For LPARs, this option distinguishes between FRUs associated with the overall system and FRUs assigned to a specific partition.
-s *serial_number*	Specifies the serial number for the system. The optional *serial_number* parameter is obsolete and should not be used. If the serial number is entered, that value will be used in the output of the command. In some cases, **lsvpd** is unable to automatically determine the serial number. In these cases, the user must supply the value in order for it to be displayed in the command output.
-t *type_model*	Specifies the type model for the system. The optional *type_model* parameter is obsolete and should not be used. If the type model is entered, that value will be used in the output of the command. In some cases, **lsvpd** is unable to automatically determine the type model. In these cases, the user must supply the value in order for it to be displayed in the command output.
-v	Produces verbose output for debugging purposes only.

Exit Status

Item	Description
0	The command completed successfully.
1	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. Output for the **lsvpd** command is similar to the following.

   Note: Portable applications should not parse this data.

```
*VC 5.0
*TM IBM,7029-6E3
*SE IBM,0110B721E
*PI 000B721E
*OS AIX 5.3.0.0
*FC ????????
*DS Platform Firmware
*YL U0.1-P1-X1/Y1
*RM 3F041029
*VK RS6K
*FC ????????
*DS System Firmware
*YL U0.1-P1-X1/Y2
*RM RG041029_d79e00_regatta
*VK RS6K
*FC ????????
*DS System VPD
*YL U0.1
*SE 10B721E
*TM 7029-6E3
```

```
*MN IBM980
*VK RS6K
*PA Y
*BR I0
*FC ????????
*DS PS CEC OP PANEL
*YL U0.1-L1
*SN YL1124350190
*EC H64013
*CC 28D3
*FN  97P3352
*DC BD 200210290851
*VK RS6K
*FC ????????
*DS 2 WAY BACKPLANE
*YL U0.1-P1
*SN YL1123354433
*PN 80P3099
*CC 26F5
*CE 1
*FN  80P3099
*VK RS6K
*FC ????????
*DS CSP
*YL U0.1-P1-X1
*SN YL1024360048
*PN 80P5573
*CC 28D0
*CE 1
*FN  80P5573
*RM 3F041029
*VK RS6K
*FC ????????
*DS IBM 1.8V VRM
*YL U0.1-P1-V1
*FN  24P6892
*VK RS6K
*FC ????????
*DS IBM 2.5V VRM
*YL U0.1-P1-V2
*FN  53P5623
*VK RS6K
*FC ????????
*DS IBM 1.2V VRM
*YL U0.1-P1-V3
*FN  53P5621
*VK RS6K
*FC ????????
*DS A IBM AC PS
*YL U0.1-V2
*SN YL1023C90045
*EC H85582
*CC 51B5
*FN  97P5101
*VK RS6K
*FC ????????
*DS IBM Air Mover
*YL U0.1-F1
*FN  53P4612
*VK RS6K
*FC ????????
*DS IBM Air Mover
*YL U0.1-F2
*FN  53P4612
*VK RS6K
*FC ????????
*DS IBM Air Mover
```

```
*YL U0.1-F3
*FN  53P4612
*VK RS6K
*FC ????????
*DS VSBPD4E1  U4SCSI
*YL U0.1-P2
*SN YL11243550F4
*PN 80P4611
*EC H85823
*CC 28D2
*FN  80P4610
*FS
*VK RS6K
*FC ????????
*DS MEDIA BACKPLANE
*YL U0.1-P4
*SN YL1124341459
*PN 80P3510
*EC H85610
*CC 28D1
*FN  80P3516
*VK RS6K
*FC ????????
*DS PCI-X Dual Channel Ultra320 SCSI Adapter
*AX sisscsia1
*PL 1Z-08
*CD 10140266
*PN 97P6513
*FN 97P6513
*SN YL11A5013461
*MN 001A
*EC 1
*RM 05080064
*Z0 5702
*YL U0.1-P1-I1
*FC ????????
*DS IDE DVD-ROM Drive
*AX cd0
*PL 1G-19-00
*MF IBM
*TM DROM00205
*RL NR38
*Z0 058002028F000010
*YL U0.1-P1-X1/Q6-A0
*FC ????????
*DS 16 Bit LVD SCSI Disk Drive
*AX hdisk0
*PL 1S-08-00-5,0
*MF IBM
*TM ST336607LC
*FN 00P3068
*RL 4335304A
*SN 000D7D3B
*EC H12094
*PN 00P2676
*Z0 000003129F00013E
*Z1 0812C512
*Z2 0002
*Z3 04341
*Z4 0001
*Z5 22
*Z6 H12094
*YL U0.1-P1/Z1-A5
*FC ????????
*DS 16 Bit LVD SCSI Disk Drive
*AX hdisk1
*PL 1S-08-00-8,0
```

```
*MF IBM
*TM ST336607LC
*FN 00P3068
*RL 4335304A
*SN 000D7996
*EC H12094
*PN 00P2676
*Z0 000003129F00013E
*Z1 0812C512
*Z2 0002
*Z3 04340
*Z4 0001
*Z5 22
*Z6 H12094
*YL U0.1-P1/Z1-A8
*FC ????????
*DS Diskette Drive
*AX fd0
*PL 01-D1-00-00
*YL U0.1-P1-X1-D1
*FC ????????
*DS Asynchronous Terminal
*AX tty0
*PL 01-S1-00-00
*YL U0.1-P1-X1/S1-L0
*FC ????????
*DS SCSI Enclosure Services Device
*AX ses0
*PL 1S-08-00-15,0
*MF IBM
*TM VSBPD4E1  U4SCSI
*RL 4610
*SN 243550F4
*Z0 0D0002022F004000
*FN 80P4610
*FL DB1
*FS
*YL U0.1-P1/Z1-Af
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M5
*SN YL10243591YT
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M7
*SN YL10243591YP
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
*YL U0.1-P1-M4
*SN YL1024359208
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
*FC ????????
*DS IBM MS 512 MB
```

```
*YL U0.1-P1-M2
*SN YL1024359204
*PN 00P5767
*CC 30D2
*FN  00P5767
*SZ 512
*VK RS6K
```

Location

/usr/sbin/lsvpd

Related information:

uname command

---

lsvsd Command

Purpose

Displays configured virtual shared disks and their characteristics.

Syntax

lsvsd [**-l** | **-s**[*vsd_name*...]] | [**-i**]

Description

The **lsvsd** command displays information about virtual shared disks currently configured on the node on which the command is run. If a list of virtual shared disks follows the flags, information about those virtual shared disks is displayed. **lsvsd** with no arguments or flags lists the names of all the virtual shared disks currently configured on the node.

The **lsvsd** command displays information about both the configuration and the usage of a virtual shared disk.

You can use the System Management Interface Tool (SMIT) to run the **lsvsd** command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Show All Managed Virtual Shared Disk Characteristics** option.

Flags

-l Lists the name of the virtual shared disk, the minor number, the state, the current server node number, and, at the server only, the major and minor number of the logical volume. (This flag is lowercase **l**, as in **l**ist.)

The state field can have one of the following values:

STP Stopped

SUS Suspended

ACT Active

An asterisk (*) in front of any of these values indicates that the virtual shared disk has been fenced from this node.

This flag is not compatible with the **-s** flag.

The *server_list* of the virtual shared disk is listed.

-s Lists usage statistics about the virtual shared disks. It lists the number of local logical read and

write operations, the number of remote logical read and write operations, the number of client logical read and write operations, the number of physical reads and writes, and the number of 512-byte blocks read and written. The number of blocks read and written is cumulative, so issue **ctlvsd -V** to reset this count before measuring it.

The local logical operations are requests which were made by a process executing at the local node, whereas the remote logical operations were made by a process executing on a remote node. *Client operations* are those local logical requests that cannot be satisfied locally, and have to be sent to a remote node. *Physical* operations are those server operations which must be passed to the underlying disk device.

This flag is not compatible with the **-l** flag.

-i Lists the "node to IP address" map that is currently used by the virtual shared disk driver.

Parameters

vsd_name
Specifies a virtual shared disk. This parameter is valid only with the **-l** and **-s** flags.

Security

You must be in the AIX **bin** group to run this command.

Restrictions

You must issue this command from a node that is online in the peer domain. To bring a peer domain online, use the **startrpdomain** command. To bring a particular node online in an existing peer domain, use the **startrpnode** command. For more information on creating and administering an RSCT peer domain, refer to *RSCT Administration Guide* .

Examples

1. To list all virtual shared disks in the system, enter:

   ```
   lsvsd
   ```

   The system displays a message similar to the following:

   ```
   vsd00

   vsd01

   .
   .
   .
   ```

2. To list virtual shared disks and their characteristics, enter:

   ```
   lsvsd -l
   ```

   The system displays a message similar to the following:

   ```
   minor  state  server  lv_major  lv_minor  vsd_name  size (MB)
   83     STP      -1        0         0      vsdn08v3    20
   84     STP      -1        0         0      vsdn08v4    16
   ```

3. To list statistics about virtual shared disks and precede the column output with a header, enter:

   ```
   lsvsd -s
   ```

   The system displays a message similar to the following:

   ```
   lc-rd lc-wt rm-rd rm-wt c-rd c-wt  p-rd  p-wt   br   bw  vsd_name
      84    84  2858   169    0    0   348   253  164  184  vsd.vsd1
       0     0     0     0    0    0     0     0    0    0  vsd.rl01
       0     0     0     0    0    0     0     0    0    0  vsd.rl02
   ```

The following table spells out the names of the headers used in the displays for the **-l** and **-s** options:

Header
Meaning

minor Virtual shared disk minor number

state State of this virtual shared disk:*active, stopped, suspended*

server Primary node for this virtual shared disk

lv major
Logical volume major number

lv minor
Logical volume minor number

vsd_name
Name of this virtual shared disk

lc-rd Local logical reads

lc-wt Local logical writes

rm-rd Remote logical reads

rm-wt Remote logical writes

c-rd Client logical reads

c-wt Client logical writes

p-rd Physical reads

p-wt Physical writes

br Blocks read

bw Blocks written

Location

/opt/rsct/vsd/bin/lsvsd

---

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

Item	Description
-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**.

lswpar Command

Purpose

Lists characteristics of workload partitions.

Syntax

Tabular Formats:

lswpar [**-b** | **-Br** | **-Bf** | **-D** | **-I** | **-M** | **-N**] [**-X**] [**-a** *fieldname* [,...]] [**-q**] [**-s** *state*] [**-t** *type*] [*wparname* ...]

Paragraph Formats:

lswpar {**-G** | **-L** | **-R** | **-S** | **-T**} [**-s** *state*] [**-t** *type*] [*wparname* ...]

Delimited Formats:

lswpar {**-c** | **-d** *delim*} [**-a** *fieldname* [,...] | **-G** | {**-b** | **-Br** | **-Bf** | **-D** | **-X -I** | **-M** | **-N**} [**-a** *fieldname* [,...]] | **-R** | **-S** | **-T**] [**-q**] [**-s** *state*] [**-t** *type*] [*wparname* ...]

Description

The **lswpar** command prints information about 1 or more specified workload partition (or all workload partitions if none are specified) to standard output.

You can filter all listings according to the following workload partition states by using the **-s** flag:

Item	Description
Defined	The workload partition has been defined by the **mkwpar** command and is ready for use, but is not active. Start workload partitions in this state with the **startwpar** command.
Loaded	The workload partition has been configured in the kernel, but processes have not yet been started. **Note:** This state is visible only to programmatic consumers that use the **lswpar** command to start a workload partition.
Active	The workload partition is running normally.
Frozen	A checkpoint operation is initiated, and the processes of the workload partition are quiesced, awaiting the storing phase. **Note:** The **Frozen** state is only visible when you use the **lswpar** command to checkpoint a workload partition. The checkpoint or restart function requires additional software package other than base WPAR.
Paused	A checkpoint or restart operation has been performed, and the processes of the workload partition are ready to be resumed or killed. The checkpoint or restart functionality requires additional software.
Maintenance	A workload partition can be put into maintenance mode with the **startwpar** command. During maintenance mode, the workload partition has been configured in the kernel and the file systems have been mounted, but processes do not start.
Moving	An asynchronous checkpoint-restart operation has been performed. Although the workload partition is **Active** on the arrival server, the workload partition appears in the **Moving** state on the departure server until all resources have been successfully transferred. The checkpoint or restart functionality requires additional software.
Transitional	An administrative operation is in progress. The workload partition is being created, started, stopped, configured, and so on.
Broken	An administrative operation failed, leaving this workload partition in an unusable state.
Error	An error occurred because of invalid elements such as workload partition name and flags.

You can filter all listings according to the following workload partition types by using the **-t** flag:

Item	Description
Application	This type is an application workload partition, running a single process (or a group of processes that are invoked by that means) without isolated system services. The process or group of processes inherits its operating environment (file systems, security, devices. and so on) from the environment where the application workload partition was created.
System	This type is a system workload partition, emulating an independent, fully functional instance of the operating system.

If additional checkpoint or restart software is installed, you can also specify the following type:

Item	Description
Checkpointable	This workload partition is enabled for checkpoint or restart functions. **Tip:** This type is not a mutually exclusive workload partition type. **Checkpointable** workload partitions are still either **System** or **Application** workload partitions.

If additional versioned workload partition software is installed, you can also specify the following type:

Item	Description
Versioned	This workload partition is running in operating system compatibility mode. **Tip:** This type is not a mutually exclusive workload partition type. **Versioned** workload partitions are still **System** workload partitions.

Versioned

Tabular Formats

If no options are used, the output is tabular as shown in the following example:

```
Name         State    Type    Hostname    Directory   RootVG WPAR
------------------------------------------------------------------
wpar name    state    type  hostname    root directory   yes/no
...            ...    ...   ...           ...              ...
```

In tabular formats, there might be multiple records per WPAR. The **-D**, **-I**, **-M**. and **-N** flags display in a tabular format, but can be combined with the **-c** and **-d** flags to generate a delimited format. You can use the **-a** flag to customize which fields are displayed in tabular formats. You can use the **-q** flag to suppress the table headers.

Tip: Do not rely on the exact format and content of tabular output for automated purposes. Delimited formats are provided for output that can be parsed.
The width of each field in a tabular format is expanded according to the longest value in that column. Therefore, the output might wrap on narrow screens, depending on the fields requested.

Paragraph Formats

In paragraph formats, each field has one value for one WPAR. You can use the **-G**, **-R**, **-S**, and **-T** flags to display paragraph-style subsets of workload partition configurations. The **-L** flag displays a long listing, which is a combination of the data that is presented by the **-D**, **-G**, **-I**, **-M, -N**, **-R**, **-S**, and **-T** flags. Otherwise, formats cannot be combined.

Delimited Formats

Delimited formatsare used to produce machine-readable formats. You can select any delimiting characters. You can generate delimited formats by using the **-c** or **-d** flag. You can use the **-a** flag to customize which fields are displayed. You can use the **-q** flag to suppress the header line. The paragraph format flags (**-G**, **-R**, **-S**, and **-T**) and tabular format flags (**-D**, **-I**, **-M**, and **-N**) can be used individually to limit the display to the corresponding predefined set of fields.

Flags

Item	Description
-a *fieldname*	Limits tabular or delimited displays to the specified 1 or more fields. Multiple field names must be separated by commas with no spaces. This flag is mutually exclusive with the **-G**, **-R**, **-S**, **-L**, or **-T** flag. By default, the display consists of 1 WPAR per line. You can specify any of the following fields: **General** • **Name** (the WPAR name) • **Cid** (the ID of aWPAR • **Key** (the key of a WPAR • **Rootvgwpar** RootVG WPAR (A yes/no value is displayed to identify whether the WPAR is a RootVG WPAR • **Uuid** (the UUID of a WPAR • **Vipwpar** VIP WPAR (A yes/no value is displayed to identify whether the WPAR is a VIP WPAR. This field only applicable for an Application WPAR • **State** • **Type** (system or application) • **Hostname** • **Routing** • **Directory** • **Privateusr** The sample output is displayed as shown here: 0> lswpar -a name,privateusr test Name Private /usr? --- test no • **Script** (user-supplied start or stop script) • **Auto** – If the value for this field is **yes**, the process is automatically started on global system restart. – If the value for this field is **no**, the process is not automatically started on global system restart. • **Application** (tracked process for application WPARs) • **Checkpointable** • **Owner** • **OStype** (a non-zero value indicates a versioned WPAR, a value of 0 or null indicates a native WPAR) **Resource Controls** • **Active** – If the value for this field is **yes**, resource controls are active – If the value for this field is **no**, resource controls are inactive • **Rset** • **Shares_CPU** • **CPU** • **Shares_memory** • **Memory** • **ProcVirtMem** • **TotalProcesses** • **TotalThreads** • **totalPTYs** • **totalLargePages** • **totalVirtmem** • **pct_msgIDs** • **pct_semIDs** • **pct_shmIDs** • **pct_pinMem**

Item **Description**

(*Fields that you can specify with the* **-a** *flag, are as follows*)

Devices

- **Name** (the name of the WPAR)
- **Devname** (the name of the device)
- **Devtype** (pseudo, disk, clone)
- **Rootvg**

The display consists of 1 device per line. The following displays sample output:

```
0> lswpar -Da name,devname,rootvg test
Name    Device Name         RootVG
------------------------------
test    hdisk1              yes

0> lswpar test
Name    State   Type    Hostname    Directory     RootVG WPAR
-------------------------------------------------------
test    D       S       test        /wpars/test   yes
```

Kernel Extensions

- **Name** (the name of the WPAR)
- **Kext** (the full path to the kernel extension)
- **Local**
- **Major**
- **kextstatus** (allocated or exported)
- **checksum** (checksum of the kernel extension)
- **mtime** (modification time of the kernel extension)

The display consists of 1 kernel extension per line.

WPAR-Specific Routes

A workload partition might have more than 1 route. So if you use the **-I** flag, you can specify the **-a** flag with the following fields:

- **name** (the WPAR name)
- **rtdest**
- **rtgateway**
- **rtinterface**
- **rttype**
- **rtfamily**

The display consists of 1 route per line.

Networks

A WPAR might have more than 1 network. So when you use the **-N** flag, you can specify the **-a** flag with the following fields:

- **Name** (the WPAR name)
- **Interface**
- **Address**
- **Netmask**
- **Broadcast**

The display consists of 1 network per line.

Item	Description
	Mounts A workload partition might have more than 1 mount. So when you use the **-M** flag, you can specify the **-a** flag with the following fields: • **Name** (the WPAR name) • **Mountpoint** (the mount point name) • **Device** (the object mounted) • **Vfs** (the virtual-file-system type) • **Nodename** (node name, if the mount is remote) • **Options** (any mount options) The display consists of 1 mount per line. **Security** • **Privs** (the list of privileges) **Operation** • **Opname** (the name of the administration operation that is being performed) • **Oppid** (the process ID of the operation) • **Opstart** (the start time of the operation) **Bootlist** When you use the **-b** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **bootlist** (an ordered list of bootsets in comma-separated format) The display consists of 1 bootlist per line. **Bootset** When you use the **-Br** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **devname** (the name of the device) • **vdevname** (the name of the virtual device) • **rootvg** • **bootset** (the bootset devices of the WPAR) The display consists of 1 bootset per line. When you use the **-Bf** flag, you can specify the **-a** flag with the following fields: • **name** (the name of the WPAR) • **mountpoint** (the name of the mountpoint) • **device** (the object mounted) • **vfs** (the type of the virtual file system) • **options** (any mount options) • **bootset** (the bootset file systems of the WPAR) The display consists of 1 bootset per line.
-b	Shows the bootlist of the workload partition. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Bootlist`
-Br	Produces detailed bootset information for each requested RootVG WPAR. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Device Name - Type - Virtual Device - RootVG - Bootset`
-Bf	Produces detailed bootset information for each requested non-RootVG WPAR. If the **-c** or **-d** flag is not specified, the output for each WPAR has the following tabular format: `Name - Mount Point - Device - Vfs - Options - Bootset`

Item **Description**

-c Produces colon-separated output suitable for machine parsing. It is mutually exclusive with the **-L** flag. The default output format (when the **-D**, **-G**, **-I**, **-M**, **-N**, **-R**, **-S**, and **-T** flags are not used) is as follows:

```
name:state:type:hostname:directory
```

The **state** field is one or more of the following valid states:

D Defined

L Loaded

A Active

F Frozen

P Paused

N Maintenance

M Moving

T Transitional

B Broken

E Error

The **type** field is one or more of the following valid types:

A Application workload partition

S System workload partition

L Versioned system workload partition

-d *delim* Produces delimiter-separated output suitable for machine parsing. It is mutually exclusive with the **-L** flag. The output format when the **-d** flag is specified is the same as with when the **-c** flag is specified, but with *delim* as the delimiter output between fields.

-D Produces detailed device information for each requested WPAR. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. If the **-c** or **-d** flag is not specified, each WPAR output has the following tabular format:

```
==============================
Name - Device Name - Type - Virtual Device - RootVG - Status
```

-G Produces detailed general setting information for each requested WPAR. It is mutually exclusive with the **-I**, **-L**, **-M**, **-D**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format:

```
===============
Name - State
===============
Type:                   {S|A}
Hostname:               HostnameWPAR
-Specific Routing:  {yes|no}
Directory:              Directory
Start/Stop Script:      /path/to/userScript
Auto Start:             {yes|no}
Private /usr:           {yes|no}
Checkpointable:         {yes|no}
Application:            /path/to/trackedProcess
Owner:
Architecture: WPAR compatibility architecture
OStype:      <i>Integer value representing operating system type<i>
Cross-WPAR IPC: {yes|no}
UUID:                   String value representing universally unique ID
```

With the **-c** or **-d** flag, the output is as follows:

```
name:state:type:rootvgwpar:hostname:routing:directory:owner:script:
auto:privateusr:checkpointable:application:ostype
```

Item	Description

-I	Produces detailed information about user-specified network routes. The **-I** flag is mutually exclusive with the **-D**, **-G**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. The **-I** flag displays only the routing table entries that are explicitly specified with the **-I** flag of the **mkwpar**, **wparexec**, or **chwpar** command. To see the full routing table for a workload partition, use the **netstat** command with the **-r** and **-@** flags. If you do not specify the **-c** or **-d** flag, tabular output is as produced as shown in the following example:
	`Name Type Destination Gateway Interface`
	`----------`
	name net|host destination gateway if
	... ...
	
	With the **-c** or **-d** flag, delimited output is produced as shown in the following example:
	`name:rttype:rtdest:rtgateway:rtinterface:rtfamily`
	
	You can use the **-I** flag with the **-a** flag to limit the output to any combination of the following fields:
	• **name** (the workload partition name)
	• **rtdest**
	• **rtgateway**
	• **rtinterface**
	• **rttype**
	• **rtfamily**

-L	Specifies long format. Produces detailed paragraph-formatted information for each requested workload partition. It is mutually exclusive with the **-c**, **-d**, **-D**, **-G**, **-I**, **-M**, **-N**, **-q**, **-R**, **-S**, or **-T** flag.
	
	If you want to parse data, do not use the **-L** output. Use the delimiter-separated forms (the **-c** or **-d** flag) for generating output that can be parsed. Each workload partition has formatted output similar to the following example:
	
	`================`
	Name - State
	`================`
	`GENERAL`
	`Type: {S|A}`
	`Hostname: `*Hostname*`WPAR`
	`-Specific Routing: {yes|no}`
	`Directory: `*Directory*
	`Start/Stop Script: `*/path/to/userScript*
	`Auto Start: {yes|no}`
	`Private /usr: {yes|no}`
	`Checkpointable: {yes|no}`
	`Application: `*/path/to/trackedProcess*
	`Owner:`
	`OStype: <i>Integer value representing operating system type<i>`
	`Cross-WPAR IPC: {yes|no}`
	`Architecture:WPAR compatibility architecture`
	`UUID: String value representing universally unique ID`
	
	`NETWORK`
	`Interface Address Mask/Prefix Broadcast`
	`----------`
	if A.B.C.D A.B.C.D A.B.C.D
	... ...
	
	`USER-SPECIFIED ROUTES`
	`Type Destination Gateway Interface`
	`----------`
	net|host destination gateway if
	... ...
	`FILESYSTEMS`
	`MountPoint Device Vfs Nodename Options`
	`----------`
	mountpoint device vfs node options
	... ...

Item	Description
	(*A long-format example by the* **-L** *flag, is as follows*)

```
RESOURCE CONTROLS
Active:                                 {yes|no}
RSet:                                   rset
CPU Shares:                             n
CPU Limits:                             m%-S%,H%
Memory Shares:                          n
Memory Limits:                          m%-S%,H%
Per Process Virtual Memory Limit:       nMB
Total Processes:                        n
Total Threads:                          n
Total PTYs:                             n
Total Large Pages:                      n
Max Message queue IDs:                  n%
Max Semaphore IDs:                      n%
Max Shared memory IDs:                  n%
Max Pinned memory:                      n%
OPERATION
Operation:  %c
Process ID: %p
Start time: %t

SECURITY SETTINGS
Privileges: privilege list
            ...

DEVICE EXPORTS
Name                  Type
Virtual Device          RootVG         Status
device name type virtual device name yes/no device status
...                                 ...
```

-M Produces detailed mount information for each requested workload partition. The file systems that are mounted from outside the workload partition are listed and the file systems that are defined within the workload partition are not included. The **-M** flag is mutually exclusive with the **-G**, **-I**, **-L**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, tabular output is produced as shown in the following example:

```
Name     MountPoint    Device    Vfs    Nodename  Options
-------------------------------------------------------------
name      mountpoint    device   vfs   node     options
...       ...          ...      ...   ...      ...
```

With the **-c** or **-d** flag, delimited output is produced as shown in the following example:

```
name:mountpoint:device:vfs:nodename:options
```

It can be used with the **-a** flag to limit the output to any combination of the following fields:

- **Name** (the workload partition name)
- **Mountpoint** (the mount point name)
- **Device** (the object mounted)
- **Vfs** (the virtual-file-system type)
- **Nodename** (node name, if the mount is remote)
- **Options** (any mount options)

Item	Description

-N	Produces detailed network information for each requested workload partition. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-R**, **-D**, **-S**, or **-T** flag. If you do not specify the **-c** or **-d** flag, tabular output is produced as shown in the following example:
	
	`Name Interface Address(6) Mask/Prefix Broadcast`
	`------------------------------------------------------------`
	`name if A.B.C.D A.B.C.D A.B.C.D`
	`...`
	`name if S:T:U:V:W:X:Y:Z R`
	`...`
	
	With the **-c** or **-d** flag, delimited output is produced as shown in the following example:
	
	`name:interface:address:mask_prefix:broadcast`
	
	You can specify the **-N** flag with the **-a** flag to limit the output to any combination of the following fields:
	• **Name** (the WPAR name)
	• **Interface**
	• **Address** (the IPv4 or IPv6 address)
	• **Mask_Prefix** (the IPv4 **netmask** field or the IPv6 **prefixlen** field)
	• **Broadcast**
	
	If a WPAR contains 1 or more name-mapped interfaces, the **lswpar** command shows only the information that is specified in the configuration file when the WPAR is in the **Defined** state. When the WPAR is in the **Active** state, the actual runtime network attributes are displayed.
	Note: When a delimited output is expected to contain IPv6 addresses, use the **-d** flag to specify an alternative delimiter because IPv6 addresses contain colons.

-q	Suppresses table headers (quiet). It is valid only for tabular and delimited output formats.

-R	Produces detailed resource control information for each requested WPAR. It is mutually exclusive with the **-G**, **-I**, **-L**, **-M**, **-N**, **-D**, **-S**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format:
	
	`===============`
	`Name - State`
	`===============`
	`Active: {yes|no}`
	`RSet:` *`rset`*
	`CPU Shares:` *`n`*
	`CPU Limits:` *`m%-S%,H%`*
	`Memory Shares:` *`n`*
	`Memory Limits:` *`m%-S%,H%`*
	`Per-Process Virtual Memory Limit:` *`n`*`MB`
	`Total Processes:` *`n`*
	`Total Threads:` *`n`*
	`Total PTYs:` *`n`*
	`Total Large Pages:` *`n`*
	`Max Message queue IDs:` *`n%`*
	`Max Semaphore IDs:` *`n%`*
	`Max Shared memory IDs:` *`n%`*
	`Max Pinned memory:` *`n%`*
	
	With the **-c** or **-d** flag, delimited output is as shown in the following example:
	
	`name:state:active:rset:shares_CPU:CPU:shares_memory:memory:`
	`procVirtMem:totalProcesses:totalThreads:totalPTYs:`
	`totalLargePages:pct_msgIDs:pct_semIDs:pct_shmIDs:pct_pinMem`

-s {[D] [L] [A] [F] [P] [N] [M] [T] [B]}	Filters the output based on workload partition states. You can use more than 1 state code. See the **-c** flag for a description of the state codes.
-S	Produces detailed security privilege information for each requested WPAR. It is mutually exclusive with the **-D**, **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, or **-T** flag. If you do not specify the **-c** or **-d** flag, each workload partition output has the following paragraph format: `==============================` *`Name - State`* `==============================` `Privileges:` *`comma-separated list of privileges assigned to the workload partition`*
-t {[A][S][C][L]}	Filters the output based on workload partition types. You can use more than 1 type code. See the **-c** flag for a description of the type codes.

Item	Description

-T	Produces detailed locking information for each requested workload partition. This flag is mutually exclusive with the **-D**, **-G**, **-I**, **-L**, **-M**, **-N**, **-R**, **-S**, or **-s** flag. It is mutually exclusive with the **-q** flag unless the **-c** flag is also specified. If you do not specify the **-c** flag, each workload partition output has the following format:
	
	`================`
	`Name - State`
	`================`
	`Operation: %c`
	`Process ID: %p`
	`Start time: %t`
	
	With the **-c** or **-d** flag, the output is as shown in the following example:
	
	`name:state:opname:oppid:opstart`

-X	Produces detailed kernel extension information for each requested workload partition in turn. It is mutually exclusive with the **-D**, **-G**, **-I**,**-L**, **-M**, **-N**, **-R**, **-S**, or **-T** flag. If the **-c** or **-d** flag is not specified, each workload partition output has the following tabular format:
	
	`Name Extension Name Local Major Status checksum`
	`---------------------------------------------------------------`
	`name /path/to/extension local major status checksum`
	`...`

wparname	Specifies 1 or more workload partitions. It must be last on the command line. It can contain shell-style wildcards to match multiple workload partition names. (In this case, use appropriate shell quotation marks to preclude shell expansion before the **lswpar** command receives the metacharacters.)

Security

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

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To view tabular information about all workload partitions, enter the following command:

```
# lswpar
Name        State  Type   Hostname            Directory     RootVG WPAR
------------------------------------------------------------------------
bar         A      S      bar.austin.ibm.com  /wpars/bar      yes
foo         D      S      foo.austin.ibm.com  /wpars/foo      no
trigger     A      A      trigger             /
```

2. To view limited tabular information about application workload partitions only, enter the following command:

```
# lswpar -t A -a name,application,script
Name     Application               Script
-------------------------------------------------------
trigger  /usr/sbin/apachectl start  /home/joe/trigger.script
```

3. To view colon-separated general information with no headers for all active and defined workload partitions, enter the following command:

```
# lswpar -G -c -q -s AD
bar:A:S:bar.austin.ibm.com:/wpars/bar:/home/bar/wpar.scr:no:no:yes::no
foo:D:S:foo.austin.ibm.com:/wpars/foo::no:no:no::no
trigger:A:A:trigger:/:/home/joe/trigger.script
:no:no:yes:/usr/sbin/apachectl start:no
```

4. To view extended information about the workload partition named `trigger`, enter the following command:

```
 # lswpar -L trigger
================
trigger - Active
================
GENERAL
Type:                A
```

```
Hostname:               triggerWPAR
-Specific Routing:  yes
Directory:              /
Start/Stop Script:      /home/joe/trigger.script
Auto Start:             no
Private /usr:           no
Checkpointable:         yes
Application:            /usr/sbin/apachectl start

NETWORK
Interface    Address          Mask/Prefix      Broadcast
------------------------------------------------------------------
en0          1.2.3.4          255.255.255.0    1.2.3.255
en1          5.6.7.8          255.255.255.0    5.6.7.255

USER-SPECIFIED ROUTES
Type        Destination   Gateway     Interface
------------------------------------------------------------------
net         9.1.2.24      1.2.3.1     en0
host        192.168.1.2   1.2.3.1     en1

FILESYSTEMS
MountPoint              Device          Vfs    Nodename  Options
------------------------------------------------------------------
/share                  /nfs2/share     nfs    nfsserver rw

RESOURCE CONTROLS
Active:                              yes
RSet:                                isp1
CPU Shares:                          2
CPU Limits:                          5%-10%,50%
Memory Shares:                       3
Memory Limits:                       10%-20%,30%
Per-Process Virtual Memory Limit:    1024MB
Total Processes:                     64
Total Threads:                       1024
Total PTYs:                          8
Total Large Pages:                   16
Max Message queue IDs:               20%
Max Semaphore IDs:                   30%
Max Shared memory IDs:               50%
Max Pinned memory:                   20%
OPERATION:
Operation:                           restart
Process ID:                          905266
Start time:                          11:19


Privileges:                          PV_AU_,PV_AU_ADD,PV_AU_ADMIN,PV_AU_PROC,
                                     PV_AU_READ,PV_AU_WRITE,PV_AZ_ADMIN,
                                     PV_AZ_CHECK,PV_AZ_READ,PV_AZ_ROOT,PV_DAC_,
                                     PV_DAC_GID,PV_DAC_O,PV_DAC_R,PV_DAC_RID,
                                     PV_DAC_UID,PV_DAC_W,PV_DAC_X,PV_DEV_CONFIG,
                                     PV_DEV_QUERY,PV_FS_CHOWN,PV_FS_CHROOT

DEVICE EXPORTS
Name                                 Type         Virtual Device    RootVG      Status
-------------------------------------------------------------------------------------
hdisk4                               disk                           yes         ALLOCATED
/dev/null                            pseudo                                     ALLOCATED
/dev/tty                             pseudo                                     ALLOCATED
/dev/random                          pseudo                                     ALLOCATED
/dev/urandom                         pseudo                                     ALLOCATED
/dev/console                         pseudo                                     ALLOCATED
/dev/zero                            pseudo                                     ALLOCATED
/dev/clone                           pseudo                                     ALLOCATED
/dev/sad                             clone                                      ALLOCATED
```

5. To view machine-readable network information that is separated by pipes for workload partitions called roy, enter the following command:

```
# lswpar -d'|' -N roy
#name|interface|address|mask_prefix|broadcast
roy|en0|192.168.1.50|255.255.255.128|192.168.1.127
roy|en1|2001:DB8::|32|
```

6. To view machine-readable, resource-control information for all workload partitions, enter the following command:

```
# lswpar -cR
#name:state:active:rset:shares_CPU:CPU:shares_memory:memory:procVirtMem:
totalProcesses:totalThreads:totalPTYs:
totalLargePages:pct_msgIDs:pct_semIDs:pct_shmIDs:pct_pinMem
dale:A:no:::::::
roy:A:yes:rogers:3::2::32:128
trigger:A:yes:isp1:2:5%-10%,50%:3:10%-20%,30%:1024MB:64:1024:8:
16:20%:30%:50%:20%
```

7. To view operation information about the workload partition named `foo`, enter the following command:

```
# lswpar –T foo
================================================================
foo - Transitional
================================================================
Operation: restart
Process ID: 905266
Start time: 11:19
```

8. To view information about devices that are exported and allocated in the workload partitions named `roy`, enter the following command:
9.

```
# lswpar -D roy
Name  Device Name      Type    Virtual Device  RootVG    Status
-----------------------------------------------------------------------
roy   /dev/null       pseudo                          EXPORTED
...
roy   fcs0            adapter                         EXPORTED
roy   hdisk2          disk      hdisk0          yes
```

10. To view information about bootset of a RootVG workload partition, enter the following command:

```
lswpar -Br <WPAR name>
```

11. To view information about bootlist of a workload partition, enter the following command:

```
lswpar -b <WPAR name>
```

Related reference:

“mkwpar Command” on page 853

Related information:

rebootwpar command

syncwpar command

syncroot command

---

luit Command

Purpose

Supports locale and ISO 2022 for Unicode terminals.

Syntax

luit [options] [--] [program [args]]

Description

The **luit** command is a filter that runs between an arbitrary application and a UTF-8 terminal emulator. The **luit** command converts application output from the locale’s encoding into UTF-8, and converts terminal input from UTF-8 into the locale’s encoding.

Note: Multilingual applications must be set to generate only the UTF-8 code. You must not use the command to use different output other than UTF-8.

The **luit** command is invoked transparently by the terminal emulator. For information on running the **luit** command from the command line, see Examples.

Options

Item	Description
-h	Displays the help summary.
-list	Lists the supported **charsets** and **encodings**.
-v	Verbose.
-c	Converts the standard input to standard output.
-x	Exits as soon as the child function dies. This might cause the **luit** command to loose data at the end of the output of the child function.
-argv0 *name*	Sets the name of the child that is passed along with the **argv[0]** command.
-encoding *encoding*	Specifies that the **luit** command use encoding rather than the current locale encoding.
+oss	Disables interpretation of single shifts in the application output.
+ols	Disables interpretation of locking shifts in the application output.
+osl	Disables interpretation of character set selection sequences in the application output.
+ot	Disables interpretation of all sequences and passes all sequences in the application output to the terminal unchanged.
-k7	Generates 7-bit characters for keyboard input.
+kss	Disables generation of single-shifts for keyboard input.
+kssgr	Uses GL codes after a single shift for keyboard input. By default, GR codes are generated after a single shift when generating 8-bit keyboard input.
-kls	Generates locking shifts (SO/SI) for keyboard input.
-gl *gn*	Sets the initial assignment of GL. The argument must be one of **g0**, **g1**, **g2**, or **g3**. The default value depends on the locale, and is usually **g0**.
-gr *gk*	Sets the initial assignment of GR. The default value depends on the locale, and is usually **g2** except for EUC locales, where it is **g1**.
-g0 *charset*	Sets the value of **charset** that is initially selected in **G0**. The default value depends on the locale, but is usually ASCII.
-g1 *charset*	Sets the value of **charset** that is initially selected in **G1**. The default value depends on the locale.
-g2 *charset*	Sets the value of **charset** that is initially selected in **G2**. The default value depends on the locale.
-g3 *charset*	Sets the value of **charset** that is initially selected in **G3**. The default value depends on the locale.
-ilog *filename*	Logs all the bytes received from the child into *filename*.
-olog *filename*	Logs all the bytes sent to the terminal emulator into *filename*.

Examples

1. To adapt an instance of XTerm to the locale's encoding, current versions of XTerm invoke the **luit** command automatically when it is needed. If you are using an older release of XTerm, or a different terminal emulator, you can invoke the **luit** command manually:

   ```
   $ xterm -u8 -e luit
   ```

2. If you are running in a UTF-8 locale but need to access a remote machine that does not support UTF-8, the **luit** command can adapt the remote output to your terminal:

   ```
   $ LC_ALL=fr_FR luit ssh legacy-machine
   ```

Files

Item	Description
`/usr/lib/X11/fonts/encodings/encodings.dir`	Contains the system-wide encoding directory.
`/usr/lib/X11/locale/locale.alias`	Contains the file mapping locales to locale encoding.

lvmo Command

Purpose

Manages lvm pbuf tunable parameters.

Syntax

lvmo -v *Name* **-o** *Tunable* [*=NewValue*]

lvmo -a [**-v** *vgname*]

lvmo -L [*Tunable*]

Description

The **lvmo** command sets or displays pbuf tuning parameters. The equal sign can be used to set a particular tunable to a given value. Otherwise, if no equal sign is used, the value of the tunable will be displayed.

Attention: Misuse of the **lvmo** command can cause performance degradation or operating-system failure.

The **lvmo -a** command generates pbuf and blocked I/O statistics. The pbuf and blocked I/O report has the following label:

Label	Description
vgname	Volume group name specified with the **-v** option.
pv_pbuf_count	The number of pbufs that are added when a physical volume is added to the volume group.
total_vg_pbufs	Current total number of pbufs available for the volume group.
max_vg_pbuf_count	The maximum number of pbufs that can be allocated for the volume group.
pervg_blocked_io_count	Number of I/O's that were blocked due to lack of free pbufs for the volume group.
pv_min_pbuf	The minimum number of pbufs that are added when a physical volume is added to any volume group.
global_blocked_io_count	Number of I/O's that were blocked due to lack of free pbufs for all volume groups.
aio_cache_pbuf_count	Current total number of pbufs available for aio_cache logical volume in the volume group.

Flags

Item	Description
-a	Displays value for all tunable parameters, one per line in pairs *tunable = value*.
-o *Tunable* [*=NewValue*]	Displays the value or sets *Tunable* to *NewValue*.

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

```
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT   TYPE
     DEPENDENCIES
------------------------------------------------------------------------------
global_blocked_io_count    0      0     n/a     0      0             S
     pervg_blocked_io_count
------------------------------------------------------------------------------
max_vg_pbufs              16384  n/a    n/a   16384    none          S
     max_vg_pbuf_count
     pv_min_pbuf
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
max_vg_pbuf_count          0      0      n/a    0      none          M
     max_vg_pbufs
     pv_min_pbuf
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
pervg_blocked_io_count     0      0      n/a    0      0             S
     global_blocked_io_count
------------------------------------------------------------------------------
pv_min_pbuf               512    512     n/a    0      none          D
     max_vg_pbufs
     max_vg_pbuf_count
     pv_pbuf_count
     total_vg_pbufs
------------------------------------------------------------------------------
pv_pbuf_count             512    512     n/a     1      16384         D
     max_vg_pbufs
     max_vg_pbuf_count
     pv_min_pbuf
     total_vg_pbufs
------------------------------------------------------------------------------
total_vg_pbufs            512    n/a     n/a     0      0             S
     max_vg_pbufs
     max_vg_pbuf_count
     pv_min_pbuf
     pv_pbuf_count
------------------------------------------------------------------------------
...
where:
    n/a means parameter not supported by the current platform or kernel
Parameter types:
    S = Static: cannot be changed
    D = Dynamic: can be freely changed
    B = Bosboot: can only be changed using bosboot and reboot
    R = Reboot: can only be changed during reboot
    C = Connect: changes are only effective for future socket connections
    M = Mount: changes are only effective for future mountings
    I = Incremental: can only be incremented
    d = deprecated: deprecated and cannot be changed
Value conventions:
    K = Kilo: 2^10       G = Giga: 2^30       P = Peta: 2^50
    M = Mega: 2^20       T = Tera: 2^40       E = Exa: 2^60
```

Tunable Parameters

Item	Description
pv_pbuf_count	The number of pbufs that is added when a physical volume is added to the volume group.
max_vg_pbuf_count	The maximum number of pbufs that can be allocated for the volume group. **Note**: The volume group must be varied off and varied on again for this value to take effect. This value does not affect rootvg.
pv_min_pbuf	The minimum number of pbufs that is added when a physical volume is added to any volume group. **Note:** Use the **ioo** command to change this value.
aio_cache_pbuf_count	The total number of pbufs that is allocated for aio_cache logical volume in the volume group.
workQ_size	The size of the hash table that is used to track I/O requests for logical volumes in the volume group.

Exit Status

This command returns zero for successful completion; otherwise it returns nonzero.

Security

You must have root authority to run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To display the value for the `pv_pbuf_count`, type the following:
   ```
   lvmo -v rootvg -o pv_pbuf_count
   ```
2. To set the `pv_pbuf_count` value to 2048, type the following:
   ```
   lvmo -v rootvg -o pv_pbuf_count=2048
   ```
3. To generate pbuf and blocked I/O statistics, type the following:
   ```
   lvmo -a
   ```

Location

/usr/sbin/lvmo

Related information:

vmo command

---

lvmstat Command

Purpose

Reports input/output statistics for logical partitions, logical volumes and volume groups. Also reports pbuf and blocked I/O statistics and allows pbuf allocation changes to volume groups.

Syntax

lvmstat { **-l** | **-v** } *Name* [**-e** | **-d**] [**-F**] [**-C**] [**-c** *Count*] [**-s**] [*Interval* [*Iterations*]]

lvmstat -v *Name* **-r** [**-L** | **-C**]

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.

Note: The **-e** flag and the **-d** flag are not applicable for the space reclamation statistics specified by the **-r** flag.

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.

Input/Output 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 have 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

Space reclaim statistics reports

The **lvmstat -r** command generates report for space reclaim statistics for physical volumes in the **volume** group. The reports have the following format:

Volume Group	Description
PV_name	Physical volume name
Reclaim	Space reclamation state. The possible state values are: **on** Space reclaim is supported for the physical volume. **off** Space reclaim is not supported for the physical volume. **suspend** Space reclaim is suspended by LVM configuration commands.
Mb_freeed	Amount of physical partition space is freed from logical volume by commands like **rmlv**, **rmlvcopy**, and **chfs** in megabytes
Mb_pending	Space reclamation pending for the physical volume space in megabytes.
Mb_success	Space reclamation requests succeeded at disk driver in megabytes.
Mb_failed	Space reclamation requests failed by the disk driver in megabytes.
Mb_reused	Free physical partition space reused for the logical volume without requesting the space reclamation in megabytes.
Mb_inprog	Amount of space reclaim request outstanding at the disk driver in megabytes.
io_count	Number of space reclaim I/O requests submitted to disk driver.

Volume Group	Description
io_failed	Number of space reclaim I/O requests failed by disk driver.
io_misalign	Number of space reclaim requests reported as misaligned by the disk driver.
Mb_misalign	Amount of space reclaim failed by the disk driver due to misalignment in megabytes.
Mb_resubmit	Amount of space reclaim resubmitted due to reclaim block not being aligned on physical partition blocks.
num_pp_free	Number of physical partitions are freed by LVM commands like **rmlv**, **rmlvcopy**, **chfs**, and so on.
Kb_blksize	Space reclaim block size reported by the disk driver for alignment purpose.

Flags

Item	Description
-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. This flag can also be used to resets the space reclaim statistics.
-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.
-L	Displays the space reclaim statistics in long listing mode.
-r	Prints the space reclaim statistics for all physical volumes in the volume group.
-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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To enable the statistics collection for volume group datavg (all the LVs in datavg are enabled), enter the following command:

   ```
   lvmstat -v datavg -e
   ```

2. To display the history of all the partitions of logical volume hd2, enter the following command:

   ```
   lvmstat -l hd2
   ```

3. To display the history of top five logical volumes of volume group uservg, enter the following command:

   ```
   lvmstat -v uservg -c 5
   ```

4. To display a continuous report at two second intervals for the logical volume ramlv, enter the following command:

   ```
   lvmstat -l ramlv 2
   ```

5. To display six reports at two second intervals for the volume group rootvg, enter the following command:

   ```
   lvmstat -v rootvg 2 6
   ```

6. To reset the counters for statistics for all the logical volumes in the volume group uservg, enter the following command:

   ```
   lvmstat -v uservg -C
   ```

7. To disable statistics collection for datalv, enter the following command:

   ```
   lvmstat -l datalv -d
   ```

8. To display statistics for space reclamation, enter the following command:
   ```
   lvmstat -v uservg –r
   ```
9. To display statistics for space reclamation in long listing mode, enter the following command:
   ```
   lvmstat -v uservg -r –L
   ```
10. To clear the statistics for space reclamation, enter the following command:
    ```
    lvmstat -v uservg -r -C
    ```

Files

Item	Description
/usr/sbin/lvmstat	Contains the **lvmstat** command.

Related information:

knlist command

/dev/kmem command

---

lvupdateInit Command

Purpose

Manages the list of entries to be added to the /etc/inittab file that is used to start the surrogate partition during the AIX Live Update operation.

Syntax

To add an entry to the /etc/inittab file in the surrogate partition, use the following syntax:

```
lvupdateInit -a [-i Identifier] { [ Identifier ] : [ RunLevel ] : [ Action ] : [ Command ] }
```

To list the entries to be added to the /etc/inittab file in the surrogate partition, use the following syntax:

```
lvupdateInit -l
```

To remove an entry from the list to be added to the /etc/inittab file in the surrogate partition, use the following syntax:

```
lvupdateInit -r -i Identifier
```

Description

The Live Update operation creates a customized root volume group (rootvg) to start the surrogate partition and prepare it for the Live Update operation. When the surrogate partition is started, it runs commands in a customized /etc/inittab file and other scripts like /etc/rc.tcpip. The /etc/inittab file is customized to exclude the services that are moved from the original partition during the mobility phase.

Some services that are running on the original partition might choose not to be check-pointed and keep running on the original partition. In such cases, this command can be used to add that service to the /etc/inittab file for the surrogate partition so that it is restarted during the Live Update operation. The /etc/liveupdate/lvup_inittab file contains a list of command entries to be added to the customized /etc/inittab file.

Parameters

Item	Description
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.
RunLevel	A 20-character parameter that defines the run levels in which the *Identifier* parameter can be processed. Each process started by the **init** command can be assigned with 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. Refer to the **mkitab** command for a list of valid actions that are recognized by the **init** command.
Command	A 1024-character field that specifies the shell command to be run. **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

Item	Description
-a	Adds an entry to the `/etc/liveupdate/lvup_inittab` file.
-i *Identifier*	Specifies an identifier for the new entry.
-r *Identifier*	Removes an entry from the `/etc/liveupdate/lvup_inittab` file.
-l	List the entries in the `/etc/liveupdate/lvup_inittab` file.

Examples

1. To add an entry to the `/etc/inittab` file that is used to start the Live Update operation in the surrogate partition, enter the following command:

   ```
   # lvupdateInit -a -i myserver  myserver:2:once:/opt/myapp/start_my_server
   ```

2. To remove an entry from the `/etc/inittab` file of the surrogate partition, enter the following command:

   ```
   # lvupdateInit -r -i myserver
   ```

3. To list the entries to be incorporated into the `/etc/inittab` file of the surrogate partition, enter the following command:

   ```
   # lvupdateInit -l
   ```

lvupdateRegKE Command

Purpose

The **lvupdateRegKE** command is used to register a command to be used to load a kernel extension on the surrogate logical partition (LPAR) during the AIX Live Update operation.

Syntax

```
lvupdateRegKE  [ -a kext_path -c command | -r kext_path | -l ]
```

Description

The Live Update operation includes an opportunity to load specific kernel extensions before the workload resumes execution on the updated surrogate LPAR. This command manages the list of kernel extensions to be loaded on the surrogate partition when it is started. This command can be used to add a kernel extension to the list along with loading and configuring the kernel extension. It also provides options to remove a kernel extension from the list or to display the list of kernel extensions. The list is placed in the `/etc/liveupdate/lvup_preload_KE` file. To be loaded at the start of the surrogate LPAR, a kernel extension must be included in that file, and must be loaded on the original LPAR when the Live Update operation starts.

Note: The command that is to be registered with the **lvupdateRegKE** command must be present in one of the following file systems: `/`, `/var`, `/usr`, `/opt`, `/tmp`. Also, the kernel extension to be loaded by this command must be present in one of these file systems.

Parameters

Item	Description
kext_path	A string of up to 1024 characters that specifies the full path of a kernel extension.
command	A string of up to 1024 characters that specifies a command to be used to load a kernel extension on the surrogate LPAR before the applications are resumed.

Flags

Item	Description
-a *kext_path*	Adds a kernel extension to the list of kernel extensions to be loaded during the Live Update operation. **Note:** If the **-a** flag is specified, the **-c** flag is required.
-c *command*	Loads the kernel extension. Command arguments can be included by double-quoting the string.
-l	Lists the set of kernel extensions that are loaded during a Live Update operation and the commands that are specified to load them.
-r *kext_path*	Removes a kernel extension from the list.

Examples

1. To list all the commands that are registered to load kernel extensions during the Live Update operation, enter the following command:

   ```
   # lvupdateRegKE -l
   ```

2. To register a command to load a kernel extension on the surrogate LPAR during the Live Update operation, enter the following command:

   ```
   # lvupdateRegKE -a /usr/lib/drivers/mykext -c "/opt/myapp/bin/load_kext -x -y"
   ```

3. To remove a command that was previously registered to load a kernel extension during the Live Update operation, enter the following command:

   ```
   # lvupdateRegKE -r /usr/lib/drivers/mykext
   ```

lvupdateRegScript Command

Purpose

The **lvupdateRegScript** command is used to register a script to be executed at notification points during an AIX Live Update operation.

Syntax

To register a script, use the following syntax:

```
lvupdateRegScript  -a -n label -s script -d { orig | surr } -P phase -p priority
```

To unregister a script, use the following syntax:

```
lvupdateRegScript  -r -n label -d { orig | surr } -P phase [ -p priority ]
```

To list all registered scripts, use the following syntax:

```
lvupdateRegScript  -l
```

Description

The valid phases that a script can be registered for are as follows:

LVUP_CHECK
Executed at the beginning of a Live Update operation. This phase ensures the executed scripts that any associated services are ready for the Live Update operation.

LVUP_PRE
Executed before the applications are frozen on the original logical partition (LPAR). This phase allows the executed scripts to save any data as required before the applications are frozen.

LVUP_PRE_KERNEL
Executed after the applications are frozen on the original LPAR and before the applications are restarted on the surrogate LPAR. This phase is executed on the original LPAR before it is executed on the surrogate LPAR.

LVUP_POST
Executed on the surrogate LPAR after the applications are restarted. These scripts are executed in the `chroot` environment.

LVUP_ERROR
Executed in case of an error during the `LVUP_CHECK` phase or any later phase of the Live Update operation.

LVUP_COMPLETE
Executed on the surrogate LPAR after the Live Update operation is complete. This phase includes deletion of the original LPAR and resetting of the surrogate LPAR's UUID. The scripts that are registered for this phase are executed in the `chroot` environment.

During the Live Update operation, before the `DR_CHECK` scripts are invoked, the scripts that are registered with the `LVUP_CHECK` phase are executed. For the phases `LVUP_CHECK`, `LVUP_PRE`, `LVUP_PRE_KERNEL`, `LVUP_POST`, and `LVUP_COMPLETE`, the scripts are executed in a specific priority order that varies between the original and the surrogate logical partitions. On the original LPAR, the order is from priority 1 to priority 10, and the order is reversed on the surrogate LPAR, executing priority 10 first and on down to priority 1. In case of an `LVUP_ERROR` event, the scripts are executed in the opposite priority order.

The same methodology is applied to rest of the phases.

The script owner must specify if the script must be run on the original or the surrogate LPAR. If the script is to be run on both the original and the surrogate LPARs, it must be registered twice, once for the original LPAR and once for the surrogate LPAR.

When a registered script is executed, it must return 0 to indicate success, or a non-zero value if it failed. The Live Update operation fails if a script fails during the `LVUP_CHECK` or `LVUP_PRE` event.

Note: The script that is to be registered with the **lvupdateRegScript** command must be present in one of the following file systems: `/`, `/var`, `/usr`, `/opt`, `/tmp`.

Parameters

Item	Description
label	A string of up to 80 characters that specifies a label, which identifies a particular script.
script	A string of up to 1024 characters that specifies a script to be executed. The string must contain the full path to the script as well as any desired arguments.
phase	A string that specifies one of a set of phases: `LVUP_CHECK`, `LVUP_PRE`, `LVUP_PRE_KERNEL`, `LVUP_POST`, `LVUP_ERROR`, and `LVUP_COMPLETE`.
priority	An integer from 1 to 10 that identifies a priority for executing the script.

Flags

Item	Description
-a	Registers a script to be executed during the Live Update operation.
-d `orig | surr`	Specifies the LPAR on which the script is to be executed: original LPAR (`orig`) or surrogate LPAR (`surr`).
-l	Lists the scripts and associated labels that are registered.
-n *label*	Specifies a label to associate with a registered script.
-P *phase*	Selects the phase when the script is to be invoked.
-p *priority*	Specifies a priority from 1 to 10. The scripts that are registered for a particular phase are executed in order from highest (1) to lowest (10) priority.
-r	Unregisters a script.
-s *script*	Specifies the script to be executed.

Examples

1. To list all the notification scripts that are registered to be invoked during the Live Update operation, enter the following command:

   ```
   # lvupdateRegScript -l
   ```

2. To register a script to execute on the original LPAR during an `LVUP_PRE` event of the Live Update operation, enter the following command:
3. To register a script to execute on the surrogate LPAR during the `LVUP_POST` event of the Live Update operation, enter the following command:

   ```
   # lvupdateRegScript -a -n putFiles -s "/opt/myapp/bin/rest_files /var/myapp/data"
                                -P LVUP_POST -p 10 -d surr
   ```

4. To remove a script that was previously registered to be invoked during the Live Update operation, enter the following command:

   ```
   # lvupdateRegScript -r -n getFiles -d orig -P LVUP_PRE
   ```

lvupdateSafeKE Command

Purpose

The **lvupdateSafeKE** command is a utility that manipulates the list of safe kernel extensions for the AIX Live Update operation.

Syntax

```
lvupdateSafeKE  [ -a kext_path | -r kext_path | -l ]
```

Description

During a Live Update operation, a new logical partition (LPAR) is dynamically created and booted with the updated AIX kernel. When the new LPAR is booted, the device drivers are reloaded while the devices are configured. The **lvupdateRegKE** command can be used to specify commands to be executed during a Live Update operation to load other kernel extensions. After the new LPAR is prepared, the active workload from the original LPAR is checkpointed and restarted from the same point on the new LPAR.

By default, any data from the kernel extensions is not checkpointed. If there is no state data in a kernel extension, reloading the kernel extension on the new LPAR does not cause any problems while the applications are checkpointed and restarted. This condition makes a kernel extension Live Update *safe*. A kernel extension can be made safe by registering scripts to be executed during the Live Update operation that either capture any necessary state information and restore it on the new LPAR, or quiesce any subsystems that are necessary to ensure that there is no state data in the kernel extension.

A kernel extension that is in the safe kernel list ensures that if it is loaded, it does not prevent a Live Update operation. If a Live Update operation fails because of a loaded kernel extension that is not in the safe list, an error with the kernel extension name is logged under the `/var/adm/ras/liveupdate/logs`

directory. Kernel extensions can also be marked safe by specifying the **SYS_LUSAFE** flag when you load the extension by using the `sysconfig()` system call. The **lvupdateSafeKE** command does not list the extensions that were marked safe by using this method.

To bypass the check for safe kernel extensions, the Live Update operation must be started with an `lvupdate.data` file that has the entry, `kext_check = no`.

Note: Any kernel extension that is to be loaded during the Live Update operation must be in one of the following five file systems: `/`, `/var`, `/usr`, `/opt`, `/tmp`.

Parameters

Item	Description
kext_path	A string of up to 1024 characters that specifies the full path of a kernel extension.

Flags

Item	Description
-a *kext_path*	Adds a kernel extension to the list that indicates that it is safe for the Live Update operations.
-l	Lists the set of kernel extensions that are specified as safe for the Live Update operation.
-r *kext_path*	Removes a kernel extension from the list.

Examples

1. To add a kernel extension to the safe list for the Live Update operation, enter the following command:
   ```
   # lvupdateSafeKE -a /usr/lib/drivers/mydev_driver
   ```
2. To list the kernel extensions that are registered as safe, enter the following command:
   ```
   # lvupdateSafeKE -l
   ```
3. To remove a kernel extension from the safe list, enter the following command:
   ```
   # lvupdateSafeKE -r /usr/lib/drivers/mydev_driver
   ```

Related reference:

"lvupdateRegScript Command" on page 593

"lvupdateRegKE Command" on page 592

lvupdateSetProcs Command

Purpose

To add, remove, or list entries in the base process list that is used for the AIX Live Update operations.

Syntax

`lvupdateSetProcs -b [ -n` *label* `-a` *command* `| -n` *label* `-r | -l ]`

Description

Base processes are not checkpointed during a Live Update operation. These processes are left unchanged on the original logical partition (LPAR), rather than being migrated to the surrogate LPAR. This command provides a mechanism to manage the list of base processes. The list of base processes is placed in the `/etc/liveupdate/lvup_BaseProcs` file.

Parameters

Item	Description
command	A string of up to 1024 characters that specifies an executable file, which includes a full path.
label	A string of up to 80 characters that specifies a label, which is associated with a particular command that executes as a base process.

Flags

Item	Description
-a *command*	Adds the command to the specified process list. The *command* specified with the **-a** flag must execute as a direct child of the `init` process.
-b	Designates the **lvupdateSetProcs** command to work on the base process list.
-l	Lists the commands from the specified process list.
-n *label*	Specifies a label to associate with a command that is being added or removed.
-r *command*	Removes the command from the specified process list.

Examples

1. To add an entry to the base process list that is used by the Live Update operation, enter the following command:

   ```
   # lvupdateSetProcs -b -n myserv -a /usr/sbin/myservice
   ```

2. To list the commands that are registered as base processes, enter the following command:

   ```
   # lvupdateSetProcs -bl
   ```

m

The following AIX commands begin with the letter *m*.

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:

© Copyright IBM Corp. 2015, 2017

Item	Description
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, otherwise returns *False*. If you do not supply *False*, its value is null. **Note:** The behavior of **ifdef** has changed to comply with the Single UNIX Specification, Version 10. The previous behavior of **ifdef** returns the value of *True* only if *Name* is defined and is not defined as 0. By default, **ifdef** works like it did before UNIX10. The UNIX 10 behavior can be obtained by setting the environment variables *XPG_SUS_ENV* to *ON* and *XPG_UNIX98* to *OFF*.
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 signed 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. **Note:** The behavior of **eval** has changed to comply with the Single UNIX Specification, Version 10. The previous behavior of **eval** evaluates its first argument as an arithmetic expression, using 32-bit unsigned arithmetic. By default, **eval** works like it did before UNIX 10. The UNIX 10 behavior can be obtained by setting the environment variables *XPG_SUS_ENV* to *ON* and *XPG_UNIX98* to *OFF*.
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.

Item	Description
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

Item	Description
-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.

Item	Description
-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:

Item	Description
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

Item	Description
/usr/ccs/bin/m4	Contains the **m4** command.

Related information:
m4 Macro Processor Overview
Commands command
as 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

Item	Description
/usr/bin/mach	Contains the System V **mach** command.

---

machstat Command

Purpose

Reports the value of the first 4 bits of the power status register.

Syntax

machstat { **-p** | **-f** }

Description

The **machstat** command returns the value of a status register. There is no standard output or error except when using the **-f** flag on CHRP hardware.

Flags

Item	Description
-f	On non-CHRP machines, returns Power Status Register bits 10–13. On CHRP machines, displays the EPOW status, EPOW modifier, and, if present, EPOW version.
-p	Returns 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

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

To see the current value of the power status register, enter:

```
machstat -p
echo $?
```

Files

Item	Description
/etc/rc.powerfail	Shuts down a system when a power failure is detected

Related information:

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

Item	Description
File	Specifies the **nroff** or **troff** file from which the **macref** command produces output containing a list cross-referencing macros.

Output Parameters

Item	Description
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

Item	Description
-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

Item	Description
/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 reference:
"mm Command" on page 867
"mvt Command" on page 924
Related information:
nroff command
troff command

mail, Mail, or mailx Command

Purpose

Sends and receives mail.

Syntax

To Read Incoming Mail

mail -e

mail -f [**-dlHNn**] [**-F**] [*FileName*]

mail [**-dlHNn**] [**-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 user ID 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.

Notes:

- Results can be unpredictable when you run multiple instances of the **mail** command on one mailbox.
- Although the command names are different, the **mail**, **Mail**, or **mailx** command provides identical functionality.

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 7.1] 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:

Item description of mailbox

Item	Description	
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

Flags Description

Item	Description
-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.
-d	Specifies the debug information associated with the users mailbox • **uid** • **user name** • **mail file folder** (the system mailbox) • **dead letter** (the system saves incomplete messages in the **dead.letter** file in the **$HOME** • **mbox** (the personal mailbox) **Note:** The message is not sent when the program is in the debug mode.
-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.
-l	Expands the `From User` field to 256 characters to handle the long user names.
-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:

Description of environment variables

Item	Description
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

Internal variables in a mail

Item	Description
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**.

Internal variables in a mail

Item	Description
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**.

Item descriptions of Bourne shell

Item	Description
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`

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 Mail command and subcommands in *Networks and communication management*.

The following list describes the Mailbox subcommands and their functions:

Mailbox subcommands

Item	Description
=	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.

Mailbox subcommands

Item	Description
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.

Mailbox subcommands

Item	Description
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.

Conditional execution of mail subcommands

Item	Description
if *Condition* **elseendif**	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.

pi command

Item	Description
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.

Item	Description
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.

pi command

Item	Description
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.

Mail editor subcommands

Item	Description
~?	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.

Mail editor subcommands

Item	Description
~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

Files

Item	Description
$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 reference:

"mail, Mail, or mailx Command" on page 605

Related information:

bellmail command

.mailrc file

Mail handling and receiving

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

Item	Description
-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:

Item	Description
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:

Item	Description
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

Item	Description
/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:

sendmail command

Mail queue concepts and tasks

---

mailstats Command

Purpose

Displays statistics about mail traffic.

Syntax

mailstats [**-C** *cfFile*] [**-c**] [**-P**] [**-f** *StatFile*] [**-o**] [**-p**]

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:

Item	Description
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

Item	Description
-C *cfFile*	Specifies use of the *cfFile* instead of the default **sendmailcf** file.
-c	Specifies use of the **submit.cf** file instead of the **sendmail.cf** 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.
-P	Outputs information in program readable mode without clearing the statistics.

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/etc/mail/statistics	Default **sendmail** statistics file.
/etc/mail/sendmail.cf	Default **sendmail** configuration file.

Related information:

sendmail command

Mail queue concepts and tasks

Displaying mailer information

Mailer statistics logs

make Command

Purpose

Maintains, updates, and regenerates groups of programs.

Syntax

make [**-D***Variable*] [**-d** *Option*] [**-e**] [**-i**] [**-j**[*Jobs*]] [**-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:

Item	Description
:	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:

Item	Description
.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.
.SCCS_GET	This special target must be specified without prerequisites. If this special target is included in a makefile, the commands associated with this special target are used to get all SCCS files that are not found in the current directory. The default commands that are used to retrieve the source files from SCCS are replaced by the commands associated with this special target. When source files are named in a dependency list, **make** treats them just like any other target. When a target has no dependencies, but is present in the directory, **make** assumes that the file is up-to-date. If, however, a SCCS file named **SCCS/s.source_file** is found for a target **source_file**, **make** additionally checks to assure that the target is up-to-date. If the target is missing, or if the SCCS file is newer, **make** automatically issues the commands specified for the **.SCCS_GET** special target to retrieve the most recent version. However, if the target is writable by anyone, **make** does not retrieve a new version.
.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:

Item	Description
.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:

Item	Description
@	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.

Values can be appended to macro values as follows:

```
macro += word ...
macro += macro1
```

The += operator when used in place of = appends the new value with 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:

Item	Description
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

Item	Description
-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.
-j[*Jobs*]	Specifies the number of parallel jobs that **make** should use to build the independent targets. The *Jobs* parameter can take any positive integral values. If *Jobs* is not specified, the **make** command does not limit the number of parallel jobs for building the main target.
-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. Targets will not be updated when this option is specified. 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:

Item	Description
0	Successful completion.
1	The target was not up-to-date.
>1	An error occurred.

Otherwise, this command returns the following exit values:

Item	Description
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
   ```

6. To use the **make** command in parallel mode with a maximum of 10 parallel jobs to be used for building the target specified in the makefile, type:

   ```
   make -j10
   ```

Files

Item	Description
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:

sh command

make Command Overview

Commands command

makedbm Command

Purpose

Makes a Network Information Services (NIS) database map.

Syntax

To Create an 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 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

Item	Description
-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

Item	Description
/var/yp/Makefile	Contains rules for making NIS maps.

Related information:

ypinit command

yppush command

Network File System (NFS) Overview for System Management

How to Create a Non-Standard Maps

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.

Item	Description
-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 reference:

"make Command" on page 618

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

Item	Description
DESC	Causes a **DESC.out** file to be created.

Parameters

Item	Description
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:

troff command

troff Font File Format

makekey Command

Purpose

Generates an encryption key.

Syntax

makekey

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

1. To generate an encryption key for input string `1234567890`, type the following:

   ```
   $ makekey
   1234567890
   ```

   Then press the Ctrl-D key. The following encryption key is generated, and the $ (shell prompt) follows immediately after the generated key on the same line:

   ```
   90y744T/NXw1U$
   ```

2. To allow **makekey** to accept input strings through pipe, type the following command:

   ```
   $ echo 1234567890 | makekey
   ```

3. To allow **makekey** to accept input strings from a file, type the following command:

   ```
   $ cat infile
   1234567890

   $ makekey < infile
   ```

   Alternatively, you can type the following command:

   ```
   $ cat infile | makekey
   ```

Related information:

crypt, encrypt, or setkey

---

makemap Command

Purpose

Creates database maps for **sendmail**.

Syntax

makemap [**-C -N -c** *CacheSize* **-d -D-e -f -l -o -r -s -t-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

Item	Description
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.

Note: Do not use **makemap** command to create the aliases data base, but **newaliases** which puts a special token into the data base that is required by **sendmail** command.

If the TrustedUser option is set in the **sendmail** configuration file and **makemap** is invoked as the root user, the generated files are owned by the specified trusted user.

Flags

Item	Description
-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.
-D	Specifies the character to use to indicate a comment (which is ignored) instead of the default of #.
-e	Allows empty values on the right side.
-f	Disables the function of folding all uppercase letters in the key to lowercase. This flag 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 ends strings in the map. This flag must match the **-N** flag in the sendmail.cf **K** line.
-o	Specifies to append to an existing file. This flag 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 operation.
-s	Ignores safety checks on maps being created. This includes checking for hard or symbolic links in world writeable directories.
-t	Specifies the delimiter to use instead of white space. This flag is also used for dumping files.
-u	Dumps or unmaps the contents of the database to standard output.
-v	Specifies that the command verbosely print its status.

Related information:

sendmail command

man Command

Purpose

Displays manual entries online.

Syntax

man [[[**-c**] [**-t**] [*section*]] | [**-k** | **-f**]] [**-F**] [**-m**] [**-M***path*] [**-r**] [**-a**] *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:

Item	Description
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:

Item	Description
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 **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 searches the paths specified through **-M** option or MANPATH environment variable for nroff directories (man?) and formatted version directories (cat?).
4. If the **man** command does not find a manual page in the **/usr/share/man/man** or **/usr/share/man/cat** or the user-specified **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 converts the HTML file into a formatted text file to fit on the display, and displays the manual page using the command described by the PAGER environment variable.
5. If the **man** command does not find a manual page in the hypertext information bases residing in the **/usr/share/man/info** directory structure, it looks for user-specified hypertext information base (through **-M** or MANPATH). The user-defined hypertext information base, should follow the following directory structure:

   ```
   BasePath[/%{ L | l }]/DocLibraryname/Section/command_or_routine_or_filename.htm
   ```

   Where:

   - `%L` represents the ISO language notation specified using the LC_MESSAGES, `%l` represents the first 2 characters of the ISO language notation specified using the LC_MESSAGES. For example, for `LC_MESSAGES=en_US` the documents can be placed in `Path/en_US` or `Path/en`.
   - `DocLibraryname` represents the name of the documentation library.
   - `Section` represents the section name, which must be one of the following:
     - `cmds` — Represents Commands Section
     - `libs` — Represents Library Section
     - `files` — Represents Files Section

   Note: If **-m** option is specified, then the search for manual pages will be done only in the order of paths specified through **-M** or the MANPATH 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:

Item	Description
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

Item	Description
-a	Display all matching entries.
-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.
-F	Display only the first matching entry.
-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	Only search in the paths specified in MANPATH or **-M**.
-M*path*	Changes the standard location where the **man** command searches for manual information. The path is a colon-separated list of paths, where the following special symbols can be used: • %D – • The default AIX paths for man pages. • %L – A locale-specific directory location corresponding to the LC_MESSAGES category of the current locale. • %l - A locale-specific directory location corresponding to the first 2 characters of the LC_MESSAGES category of the current.
-r	Searches remotely for the manual information. If for any reason the remote search fails, then **man** performs a local search for the requested man page. Any of the following conditions can cause the remote search to fail: • The remote machine is not reachable. • There is a problem reading the URL. • A Java™ applet is not installed or it is not found in the user's search path, specified in the **PATH** environment variable. **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:

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To display information about the **grep** command, enter the following command:

   ```
   man grep
   ```

2. To display information about the **rpc_$register** library routine, enter the following command:

   ```
   man rpc_\$register
   ```

3. To display all entries in the **/usr/share/man/whatis** keyword database that contain the "mkdir" string , enter the following command:

   ```
   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 the following command:

   ```
   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 the following command:

   ```
   man -M/usr/share/man:/usr/share/man/local ftp
   ```

6. To display all matching entries, enter the following command:

   ```
   man –a title
   ```

7. To display only the first matching entry, enter the following command:

   ```
   man –F title
   ```

8. To search only in the paths specified in MANPATH or **–M**, enter the following command:

   ```
   man -m –M PATH title
   ```

9. To search in the user-defined PATH, enter the following command:

   ```
   man –M PATH title
   ```

Files

Item	Description
/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:

apropos command

catman command

whatis command

whereis command

manage_disk_drivers Command

Purpose

Displays information about storage families and the driver that manages each family or changes the driver that manages a storage family.

Syntax

manage_disk_drivers [**-l**]

manage_disk_drivers [**-d** [*device*]] [**-o** [*driver_option*]

manage_disk_drivers [**-h**]

Description

The **manage_disk_drivers** command displays a list of storage families and the driver that manages or supports each family. A storage family is a storage model. All disks within the family must be managed by the same driver.

There are two types of drivers supported: multipath I/O (MPIO) and non-MPIO. Most users select the MPIO version of the driver (`AIX_AAPCM` or `AIX_APPCM`), which is the default behavior. However, there are instances where a third-party mulitpath driver is installed. In this case, you might want to disable the built-in AIX MPIO feature. You can perform this operation by selecting the `AIX_non_MPIO` option as the driver option.

Note: Not all the storage families support the `AIX_non-MPIO` driver.

If you specify the **-d** flag with the storage device name, the **manage_disk_drivers** command changes the driver selection to the alternate supported driver that is specified with the **-o** flag.

The **manage_disk_drivers** command saves the Object Data Manager (ODM) database that reflects the driver change, and displays a message to restart the system for the change to take effect.

Flags

Item	Description
-l	Lists all the storage devices and their supported drivers.
-d	Specifies the storage device family when you are selecting a driver. **Note:** The **-o** flag is required to use this flag.
-o	Specifies the driver to be used for the selected storage family. **Note:** The **-d** flag is required to use this flag.

NO_OVERRIDE driver option

If the present driver attribute is set to NO_OVERRIDE, the AIX operating system selects an alternate path control module (PCM), such as Subsystem Device Driver Path Control Module (SDDPCM), if it is installed.

Examples

1. To list all the storage families and their supported drivers, enter the following command:

   ```
   manage_disk_drivers -l
   ```

   The system displays a message similar to the following example:

```
Device           Present Driver      Driver Options
2810XIV          AIX_AAPCM           AIX_AAPCM,AIX_non_MPIO
DS4100           AIX_APPCM           AIX_APPCM
DS4200           AIX_APPCM           AIX_APPCM
DS4300           AIX_APPCM           AIX_APPCM
DS4500           AIX_APPCM           AIX_APPCM
DS4700           AIX_APPCM           AIX_APPCM
DS4800           AIX_APPCM           AIX_APPCM
DS3950           AIX_APPCM           AIX_APPCM
DS5020           AIX_APPCM           AIX_APPCM
DCS3700          AIX_APPCM           AIX_APPCM
DS5100/DS5300    AIX_APPCM           AIX_APPCM
DS3500           AIX_APPCM           AIX_APPCM
XIVCTRL          MPIO_XIVCTRL        MPIO_XIVCTRL,nonMPIO_XIVCTRL
2107DS8K         NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
IBMFlash         NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
IBMSVC           NO_OVERRIDE         NO_OVERRIDE,AIX_AAPCM,AIX_non_MPIO
```

2. To modify the driver to use the AIX_non_MPIO option that manages 2810XIV device, enter the following command:

   ```
   manage_disk_drivers -d 2810XIV -o AIX_non_MPIO
   ```

   The system displays a message similar to the following example:

   ```
   # manage_disk_drivers -d 2810XIV -o AIX_non_MPIO
   ********************** ATTENTION ************************
   For the change to take effect the system must be rebooted
   ```

Related reference:

"mpio_get_config Command" on page 899

Related information:

Multiple Path I/O

---

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:

Item	Description
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:

Item	Description
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:
enscript command
psrev 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:

Item	Description
-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.

Item	Description
-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 reference:
"man Command" on page 631
Related information:
cw command
eqn command
nroff 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

Item	Description
-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. You cannot use **new** as a sequence name.
-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 **new** New messages 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:

Item	Description
Current-Folder:	Specifies the default current folder.

The following entry is found in the **$HOME/.mh_profile** file:

Item	Description
Path:	Specifies the MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
$HOME/.mh_profile	Specifies the MH user profile.
/usr/bin/mark	Contains the **mark** command.

Related information:

pick command

mh_alias command

mh_profile command

Mail applications

---

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.

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

Item	Description
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:

Item	Description
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

Item	Description
/dev/tty*	Supports the controlling terminal interface.
$HOME/.profile	Controls startup processes and daemons.

Related information:

sendmail command

talk command

write command

Communications and networks

---

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

Item	Description
-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:

Item	Description
`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

Item	Description
$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:

next command

prev command

mhl format file

Mail applications

---

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

Item	Description
-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

Item	Description
/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/mhmail	Contains the **mhmail** command.

Related reference:

"inc Command" on page 36

Related information:

post command

mh_profile file

Mail applications

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

Item	Description
+*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:

Item	Description
`Current-Folder:`	Sets the default current folder.
`Path:`	Specifies a user's MH directory.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
$HOME/.mh_profile	Defines the user's MH profile.
/usr/bin/mhpath	Contains the **mhpath** command.

Related information:

folder command

mh_alias command

mh_profile command

Mail applications

---

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:

1. You must consider the partition usage, reported by **lvmstat**, on the other active concurrent nodes in case of a concurrent volume group.
2. Strictness and upper bound settings are not enforced when using **migratelp**.
3. Running this command on an active, firmware-assisted, dump logical volume temporarily changes the location of the dump device to **/dev/sysdumpnull**. After you have successfully migrated the logical volume, this command calls the following command to set the firmware-assisted, dump logical volume to the original logical volume.

   ```
   sysdumpdev -P
   ```

The **migratelp** command fails to migrate partitions of striped logical volumes.

Security

To use **migratelp**, you must have root user authority.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/sbin	Directory where the **migratelp** resides.

Related reference:

"lslv Command" on page 452

"lssec Command" on page 525

Related information:

Logical volume storage

---

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.

Note:

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.
3. Running this command on a physical volume that has an active, firmware-assisted, dump logical volume temporarily changes the location of the dump device to **/dev/sysdumpnull**. After you have successfully migrated the logical volume, this command calls the following command to set the firmware-assisted, dump logical volume to the original logical volume.

   ```
   sysdumpdev -P
   ```

The allocation of the new physical partitions follows the policies defined for the logical volumes that contain the physical partitions being moved.

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 System Management Interface Tool (SMIT) **smit migratepv** fast path to run this command.

Flags

Item	Description
-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

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

Related information:

cplv command

Migrating the contents of a physical volume

Logical volume storage

System Dump Facility

---

migwpar Command

Purpose

After an operating system migration of a global system to version 7, the **migwpar** command is used to migrate a workload partition (WPAR) that was created on the version 6 global system, to version 7.

The **migwpar** command can also be used to migrate from a 5.2 or 5.3 versioned WPAR to a native version 7 WPAR.

The **migwpar** command can also be used to enable a 5.2 or 5.3 versioned WPAR to work after the global system is migrated to a new operating system level. The versioned WPAR remains at the previous level.

Syntax

migwpar [**-d**<*software_source*>] [-V]{ **-A** | **-f** <*wparNamesFile*> | [**-C**] **wpar_name** }

Description

After an operating system migration of a global system to version 7, the **migwpar** command is used to migrate a workload partition (WPAR) that was created on the version 6 global system, to version 7. Software that no longer exists (it might be replaced by a different software package) on the global system, is removed.

Ensure that all software on the global system has been migrated before you begin to migrate the WPARs. The use of the pre_migration script before the global system migration, and the post_migration script after the migration, will provide data that can be used to verify the migration, such as listing software that will be removed during the migration, and software that did not migrate.

The **-C** flag is used to migrate from a 5.2 or 5.3 versioned WPAR to a native version 7 WPAR. The **-C** flag is incompatible with the list flags (**-A** or **-f**). When you migrate a versioned WPAR, the **-d** software_source information is mandatory.

The **-V** flag can be used to enable a versioned WPAR after the global system is migrated. This option maintains the WPAR at its current level and allows the WPAR to work on the newly migrated global system. This functionality is available on IBM AIX 7.2 with Technology Level 2 or later. Only `rootvg` versioned WPARs require enablement by using the **-V** option.

Note: It is highly recommended to back up the WPAR before you begin to migrate.

A log of all actions of the **migwpar** command is saved in the `/var/adm/ras/migwpar.log` file. The output of the actual software migration of each WPAR is saved in `/var/adm/ras/devinst.log` within the WPAR.

Flags

Item	Description
-A	Migrates all migratable WPARs
-f *wparNamesFile*	Migrates the list of WPARs contained in the file wparNamesFile, one per line.
-C *wparName*	Migrates the specified 5.2 or 5.3 versioned WPAR.
-d *software_source*	Specifies the installation location that is used for the detached WPAR migration. **Note:** The install (or update) images in the specified location must be the same as the ones used to install (or update) the global system.
-V	Enables a WPAR after the global system is migrated. The **-V** option is not compatible with the **-C** or **-d** flags.

Security

Access control: Only the root user can run this command.

Examples

1. After the base operating system of GLOBAL has been migrated to version 7 to migrate a single WPAR, wpar1 to version 7, enter the following command:
   ```
   # migwpar wpar1
   ```
2. To migrate a detached WPAR, wpar2 by using install images from **/images**, enter the following command:
   ```
   # migwpar -d /images wpar2
   ```
3. To migrate all shared WPARs, enter the following command:
   ```
   migwpar -A
   ```
4. To migrate all detached WPARs by using install images in **/images**, enter the following command:
   ```
   migwpar -A -d /images
   ```
5. To migrate a 5.2 versioned WPAR wpar_52 by using install images in **/images**, enter the following command:
   ```
   migwpar -d /images -C wpar_52
   ```
6. To enable all versioned WPARs on a system after a global system is migrated, enter the following command:
   ```
   migwpar -VA
   ```

Related information:

Migrating AIX

syncwpar command

---

mirrorvg Command

Purpose

Mirrors all the logical volumes that exist on a given volume group.

Syntax

mirrorvg [**-S** | **-s**] [**-Q**] [**-c** *copies*] [**-m**] [**-p** *copyn*=*mirrorpool*] *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.

Restrictions:

- To use this command, you must either have root user authority or be a member of the **system** group.
- You cannot use the **mirrorvg** command on a snapshot volume group.
- You cannot use the **mirrorvg** command on a volume group that has an active firmware assisted dump logical volume.

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 System Management Interface Tool (SMIT) **smit mirrorvg** fast path to run this command.

Flags

Item	Description
-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.
-p *copyn=mirrorpool*	Assigns mirror pools to the copies being created. A mirror pool is assigned to a copy using the *copyn=mirrorpool* parameter. Specify a mirror pool for each copy. To specify more than one *copyn=mirrorpool* pair, provide multiple **-p** *copyn=mirrorpool* flags.
-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**:

Item	Description
rootvg mirroring	When the **rootvg** mirroring has completed, you must perform two additional tasks: **bosboot** and **bootlist**. 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.
non-rootvg mirroring	When this volume group has been mirrored, the default command causes Quorum to be deactivated.
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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 a second and third copy of the logical volumes within `datavg`, where the physical partition maps on each disk match each other exactly, enter:

   ```
   mirrorvg -m -c 3 datavg hdisk2 hdisk3
   ```

   The logical partitions in the logical volumes held on `datavg` now have three copies.

Files

Item	Description
/usr/sbin	Directory where the **mirrorvg** command resides.

Related reference:
"mklvcopy Command" on page 745
Related information:
unmirrorvg command
syncvg command
Logical volume storage
PowerHA SystemMirror Administration Guide

mirscan Command

Purpose

Search for and correct physical partitions that are stale or unable to perform I/O operations.

Syntax

mirscan -v *vgname* | **-l** *lvname* | **-p** *pvname* | **-r** *reverse_pvname* [**-a**] [**-o**] [**-q** *nblks*] [**-c** *lvcopy*] [**-s** *strictness*] [**-u** *upperbound*]

Description

The **mirscan** command examines each allocated partition on the specified device. A report is generated that lists whether the partition is stale or fresh, and lists whether it is capable of performing I/O operations. The LVM device driver is queried to determine whether the partition is stale or fresh. Regardless of whether the partition is stale or fresh, it is read to determine whether it is capable of performing I/O operations. By default the entire partition is read, but if the **-q** flag is specified, the *nblks* value determines how much of the partition will be read. If the **-a** flag is not specified, the report is printed and execution ends after all partitions are read.

If the **-a** flag is used, corrective action is taken after all the partitions have been examined. Stale partitions will be synced. If a partition is not capable of performing I/O, **mirscan** attempts to trigger bad block relocation or hardware relocation with a forced sync operation, which should write a good copy of the data to the block that is incapable of performing I/O operations. If the partition is still unreadable, the **mirscan** command attempts to migrate that partition to a new location. By default, the new location that is selected adheres to the strictness and upperbound policies for the logical volume that contains the partition. Using the **-s** flag causes the strictness value specified on the command line to override the natural strictness value of the logical volume that contains the partition. Similarly, using the **-u** flag causes the upperbound value specified on the command line to override the natural upperbound value of the logical volume that contains the partition.

The **mirscan** command prints (to standard output) a status report for the partitions scanned. If the **-a** flag is specified, the **mirscan** command also prints (to standard output) a status report containing each corrective action that is taken. If the **-o** flag is specified, the report will be in colon-separated output format. If the **-o** flag is not specified, the default behavior is to print the report in human-readable format.

Partitions on nonmirrored logical volumes are scanned and included in all reports, but no sync or migration operation is possible for such partitions. Partitions on striped logical volumes can be synced but cannot be migrated. Partitions on paging devices cannot be migrated, because this would result in a system hang if the **mirscan** process were to be paged out. Partitions on the boot logical volume cannot be

migrated. Partitions on an active firmware-assisted dump logical volume cannot be migrated. An informative error message is generated in the corrective action report for each of the preceding cases.

By default, the **mirscan** command does not take any lock on the volume group. This should allow the **mirscan** command to run in the background without interfering with other **lvm** commands. If the **-a** flag is specified and there are partitions that need to be migrated, the volume group is locked, all the migration operations are performed, and the volume group lock is released. Therefore, if the **-a** flag is specified, the impact to other **lvm** commands is minimized because the volume group is only locked during the migration operations, which are all performed at once just before the end of execution.

Flags

Item	Description
-a	Specifies that corrective action should be taken.
-c *lvcopy*	Identifies a particular copy of the logical volume. The **-c** flag can only be specified in conjunction with the **-l** flag. The **-c** flag is ignored if it is used in conjunction with the **-p**, **-r**, or **-v** flag.
-l *lvname*	Specifies the logical volume to be scanned.
-o	Specifies colon-separated output format should be used for the report. If this option is not used, the default behavior is to print a report in human-readable format.
-p *pvname*	Specifies the physical volume to be scanned.
-q *nblks*	Specifies which portions of the partition should be read. If the *nblks* value is `0`, only the first, middle, and last 512 bytes of each partition are read to determine whether the partition is capable of performing I/O operations. A nonzero *nblks* value indicates that only the first *nblks* 512 byte blocks of each partition should be read to determine whether the partition is capable of performing I/O operations. If the **-q** flag is not specified, the entire partition is read.
-r *reverse_pvname*	Specifies that any partitions in the volume group should be scanned if they do not reside on *pvname* but they do have a mirror copy on *pvname*. This could be run prior to removing *pvname* from the system, in case *pvname* somehow has the last good copy of a partition.
-s *strictness* (`y`, `n`, `s`)	Specifies a strictness value that should override the natural strictness value. Legal values are `y`, `n`, and `s`, where `y` enables strictness, `n` disables strictness, and `s` enables "superstrictness." By default, when **mirscan** has to perform a migration operation on a partition it will adhere to the natural strictness value of the logical volume that contains that partition. If the **-s** flag is used, the override strictness value will be used. If the **-s** flag is used in conjunction with the **-p**, **-r**, or **-v** flags, the override strictness value could override the natural strictness of multiple logical volumes.
-u *upperbound*	Specifies an upperbound value that should override the natural upperbound value. The upperbound value should be between 1 and the total number of physical volumes in the volume group. By default, when **mirscan** has to perform a migration operation on a partition it will adhere to the natural upperbound value of the logical volume that contains the partition. If the **-u** flag is used, the override upperbound value will be used. If the **-u** flag is used in conjunction with the **-p**, **-r**, or **-v** flags, the override upperbound value could override the natural upperbound value of multiple logical volumes.
-v *vgname*	Specifies the volume group to be scanned.

Exit Status

An exit code of `0` indicates that **mirscan** was able to complete its execution and was able to correct any error conditions that were encountered along the way. An exit code of `1` indicates that **mirscan** was able to complete its execution, but it was unable to correct every error that it found; further corrective action is still required. For example, if corrective actions would be required but the **-a** flag was not specified, an exit code of `1` is used. An exit code of `2` indicates that **mirscan** was unable to complete its execution. For example, if the target device is not listed in the ODM, an exit code of `2` is used.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To scan logical volume lv33, report the status of each partition, and have every block of each partition read to determine whether it is capable of performing I/O operations, type:

   ```
   mirscan -l lv33
   ```

2. To scan logical volume lv33, report the status of each partition, and have only the first two blocks of each partition read to determine whether it is capable of performing I/O operations, type:

   ```
   mirscan -l lv33 -q 2
   ```

3. To scan logical volume lv33, report the status of each partition, sync any stale partitions found, and migrate any partitions that are not capable of performing I/O operations, type:

   ```
   mirscan -l lv33 -a
   ```

4. To scan every allocated logical partition on hdisk4 and report the status of each partition, type:

   ```
   mirscan -p hdisk4
   ```

5. To find every allocated partition in the volume group that resides on hdisk4, and scan and report the status of all partitions that do not reside on hdisk4 but are mirror copies of a partition that resides on hdisk4, type:

   ```
   mirscan -r hdisk4
   ```

   This would be useful to run before removing hdisk4 from the system.

6. To scan volume group vg05, report the status of each allocated partition, and have the first, middle, and last 512 bytes of each partition read to determine whether that partition is capable of performing I/O operations, type:

   ```
   mirscan -v vg05 -q 0
   ```

Restrictions

Unmirrored partitions and striped partitions are not eligible for migration. Partitions on paging devices will not be migrated by mirror scan because it would result in a system hang if the **mirscan** process happened to get paged out. Partitions from the boot logical volume cannot be migrated.

Location

/usr/sbin/mirscan

Standard Output

Each line in the report corresponds to an operation on a physical partition. There are 4 types of operation that **mirscan** can perform. A *scan operation* determines whether the partition is synced and whether it is capable of performing I/O operations. A *resync operation* is a corrective action performed on stale partitions that attempts to return them to `synced` state. A *force resync operation* is a corrective action performed on partitions that are not capable of performing I/O operations, in an attempt to trigger bad block relocation or hardware relocation. At the end of the force resync operation, the partition is read again to determine whether it is capable of performing I/O operations. A *migration operation* is a corrective action performed on partitions that are not capable of performing I/O operations, in an attempt to move the data to a physical location that is capable of performing I/O.

The default format for the reports contains the following column headings. If the *-o* flag is specified, no header is displayed and the output report is printed in colon-separated output format. The columns and their meanings are as follows:

Item	Description
OP	The valid values for this field are **s**, **r**, **f**, and **m**. A value of **s** signifies a scan operation. A value of **r** signifies a resync operation. A value of **f** signifies a force resync operation, which is performed in an effort to trigger bad block relocation or hardware relocation. A value of **m** signifies a migration operation.
STATUS	The valid values for this field are **SUCCESS** or **FAILURE**. For a scan operation, FAILURE is indicated if the partition being scanned is stale or incapable of performing I/O. For a resync operation, **FAILURE** is indicated if the partition was not synchronized. For a force resync operation, **FAILURE** is indicated if the partition is still incapable of performing I/O operations. For a migration operation, **FAILURE** is indicated if the migration operation was not completed.
PVNAME	Identifies the name of the physical volume where the partition being operated on resides. For a migration operation, PVNAME refers to the source physical volume and TARGETPV refers to the destination physical volume.
PP	Identifies the physical partition number of the partition being operated on. The first partition on a particular physical volume has a PP value of `1`, not `0`.
SYNC	The valid values for this field are `synced` or `stale`. The value indicated refers to the state of the partition after the operation has been completed. For example, if a resync operation succeeds, a value of `synced` will be displayed.
IOFAIL	The valid values for this field are `yes` or `no`. The value indicated refers to the state of the partition after the operation has been completed. For example, if a migration operation succeeds then a value of `no` is displayed to indicate that the partition no longer has a problem performing I/O operations.
LVNAME	Identifies the name of the logical volume where the partition being operated on resides.
LP	Identifies the logical partition number of the partition being operated on. The first partition on a particular logical volume has an LP value of `1`, not `0`.
CP	Identifies the logical copy number of the partition being operated on. The first logical copy of a logical volume has a CP value of `1`, not `0`.
TARGETPV	Identifies the name of the physical volume that was used as the target for a migration operation. For any type of operation other than a migration operation, this field is left blank.
TARGETPP	Identifies the physical partition number of the partition that was used as the target for a migration operation. For any type of operation other than a migration operation, this field is left blank. The first partition on a particular physical volume has a TARGETPP value of `1`, not `0`.

Related information:

replacepv Command

---

mkauth Command

Purpose

Creates a new user-defined authorization.

Syntax

mkauth [**-R** *load_module*] [*Attribute* **=** *Value* ...] *Name*

Description

The **mkauth** command creates a new user-defined authorization in the authorization database. You can create authorization hierarchies by using a dot (.) in the *Name* parameter to create an authorization of the form *ParentAuth.SubParentAuth.SubSubParentAuth...*. All parent elements in the *Name* parameter must already exist in the authorization database before the new authorization is created. The maximum number of parent elements that you can use to create an authorization is 8.

If the system is configured to use multiple domains for the authorization database, the new authorization is created in the first domain specified by the **secorder** attribute in the authorizations stanza of the **/etc/nscontrol.conf** file. Use the **-R** flag to create an authorization in a specific domain.

Authorization attributes can be set at creation time through the *Attribute* **=** *Value* parameter. Every authorization that you create must have a value for the **id** authorization attribute. If you do not specify the value using the **mkauth** command, the command automatically generates a unique ID for the authorization. If you specify an ID, the value must be unique and greater than 10000.

Restriction: Authorization IDs less than 10000 are reserved for system-defined authorizations
.

When the system is operating in enhanced Role Based Access Control (RBAC) mode, modifications made to the authorization database are not used for security considerations until the database is sent to the kernel security tables using the **setkst** command. Authorizations created in the authorization database can be assigned to roles immediately, but do not take effect until the kernel security tables is updated.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for authorization creation.

Parameters

Item	Description
Attribute **=** *Value*	Initializes an authorization attribute. Refer to the **chauth** command for valid attributes and values.
Name	Specifies a unique authorization name string. **Restrictions on Creating Authorization Names:** The *Name* parameter that you specify must be unique, and can be a maximum of 63 single-byte printable characters. Although the **mkauth** command supports multibyte authorization names, authorization names to characters are restricted within the POSIX portable file name character set. The authorization name that you specify cannot begin with `aix.` because that is the designated top-level parent for system-defined authorizations and the **mkauth** command only creates user-defined authorizations. Authorization names must not begin with a dash (-), a plus sign (+), an at sign (@), a tilde (~), or contain any space, tab or newline characters. You cannot use the keywords **ALL**, **default**, **ALLOW_OWNER**, **ALLOW_GROUP**, **ALLOW_ALL**, or an asterisk (*) as an authorization name. Additionally, do not use any of the following characters within an authorization string: • : (colon) • " (quotation mark) • # (number sign) • , (comma) • = (equal sign) • \ (backslash) • / (forward slash) • ? (question mark) • ' (single quotation mark) • ` (grave accent)

Security

The **mkauth** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.auth.create	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/authorizations	rw

Examples

1. To create a top-level authorization `custom` and have the **mkauth** command assign an appropriate ID value, use the following command:
   ```
   mkauth custom
   ```
2. To create a `custom.test` child authorization and assign an ID and default description, use the following command:
   ```
   mkauth id=12000 dfltmsg="Test Authorization" custom.test
   ```
3. To create the `custom` authorization in LDAP, use the following command:
   ```
   mkauth -R LDAP custom
   ```

Related reference:

"lsauth Command" on page 363

Related information:

getauthattr command

/etc/security/authorizations file

RBAC command

---

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.

Syntax

mkboot -d *Device* [**-b**] [**-D**] [**-c**] [**-h**] [**-i**] [**-I**] [**-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.

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

Item	Description
-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.
-I (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:

Item	Description
`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

Item	Description
/usr/include/sys/bootrecord.h	Specifies the structure of the boot record.

Related reference:
"locktrace Command" on page 247
Related information:
bosboot command
Boot process
Logical volume storage

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

Item	Description
-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

Item	Description
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

Item	Description
/usr/sbin/mkC2admin	Contains the mkC2admin 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

Item	Description
-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

Item	Description
/usr/bin/mkcatdefs	Contains the **mkcatdefs** command.

Related information:
dspcat command
gencat command
runcat command
catgets command

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

Item	Description
-a *address*	Use address as the IP address of *hostname*.
-m	Configure the host as the administrative master.

Parameters

Item	Description
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

Item	Description
/usr/sbin/mkCCadmin	Contains the mkCCadmin command.

---

mkcd Command

Purpose

Creates multi-volume CDs from a **mksysb**, **savevg**, or **savewpar** backup image.

Syntax

mkcd -r *directory* | **-d** *cd_device* | **-S [-m** *mksysb_image* | **-M** *mksysb_target* | **-s** *savevg_image* | **-v** *savevg_volume_group* | **-w** *savewpar_image* | **-W** *wparname*] [**-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*] [**-f** *wparspecificationfile*] [**-e**] [**-P**] [**-l** *package_list*] [**-L**][**-b** *bundle_file*] [**-z** *custom_file*] [**-D**] [**-U**] [**-Y**] [**-n**] [**-a**] [**-A**] [**-c**] [**-Z**] [**-G** |**-N**] [**-x** file] [**-T**]

Description

The **mkcd** 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 creates a volume group backup image (**savevg**) to CD-R from a user-specified volume group or from a previously created **savevg** image. It also creates the backup image of a workload partition (**savewpar**) to CD or DVD from a user-specified workload partition or from a previously created **savewpar** image.

Note: If the system has a **multibos** environment where both instances are mounted, the only way to restore the backup is by using the **alt_disk_mksysb** command.

For DVD media, system backups that are 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 does 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 you go beyond the capacity of the media.

When a bootable backup of a root volume group is created, the boot image reflects the currently running kernel. If the current kernel is the 64-bit kernel, the backup boot image is also 64 bit, and it boots 64-bit systems only. If the current kernel is a 32-bit kernel, the backup boot image is 32 bit, and it can boot both 32-bit and 64-bit systems.

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 that is 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** command to do these tasks. The code is called by using 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, the **mkcd** command creates the necessary file systems and removes them when the command finishes executing. File systems that you supply are checked for adequate space and write access.

Note:

1. While the **mkcd** command is running, ensure that system activity is minimal.
2. If the **mkcd** command creates file systems in the backup volume group, they are excluded from the backup.

If you want to create multi-volume CDs because the volume group image does not fit on one CD, the **mkcd** command gives instructions for CD replacement and removal until all the volumes have been created.

Flags

Item	Description
-a	Does not back up extended attributes or NFS4 ACLs.
-A	Backs up DMAPI file system files.
-b *bundle_file*	Gives the full path name of the file that contains 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.
-B	Prevents the **mkcd** command from adding boot images (non-bootable CD) to the CD. Use this flag if you want to create a **mksysb** CD that you will not boot. Before you install 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 Notes for details.
-c	Does not compress or pack files as they are backed up.
-C *cd_fs_dir*	Specifies the file system that is used to create the CD file system structure, which must have at least 645 MB of available disk space (up to 4.38 GB for DVD sized images). The CD image consumes only as much room as is necessary to contain all the data on the CD. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory exists, the **mkcd** command uses that directory. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory does not exist, the **mkcd** command creates the file system **/mkcd/cd_fs** and removes it when the command finishes running. The command creates the file system in the volume group that is indicated with the **-V** flag, or **rootvg** if that flag is not used. Each time that you invoke the **mkcd** command, a unique subdirectory (that uses the process id) is created under the **/mkcd/cd_fs** directory, or in the directory that is specified with the **-C** flag. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.

Item	Description
-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.
-D	Turns on the debug output information feature. The default is no debug output.
-e	Excludes the files and directories from the backup image that is listed in the **/etc/exclude.***volume_group* file. You cannot use this flag with the **-m** or **-s** flags.
-f *wparspecificationfile*	Specifies the user-supplied WPAR specification file. This specification file of workload partition takes precedence over the **wpar.spec** file in the **savewpar** image. If you do not use the **-f** flag, the **mkcd** command restores the **wpar.spec** file from the specified **savewpar** image, or generates a new **wpar.spec** file during the creation of **savewpar**.
-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 specified **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.
-I *cd_image_dir*	Specifies the directory or file system where the final CD images are stored before they are written 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 that is given with the **-V** flag, or in **rootvg** if that flag is not used. If the **mkcd** command 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 you create multi-volume CDs. The CD image consumes only as much room as is necessary to contain all the data on the CD. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-l *package_list*	Specifies the file that contains 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 that is named with the **-p** flag. If you use the **-l** flag, you must also use the **-p** flag.
-L	Creates final CD images that are DVD sized (up to 4.38 GB).
-m *mksysb_image*	Specifies a previously created **mksysb** image. If you do not specify the **-m** flag, the **mkcd** command calls **mksysb**. For more information about where the **mksysb** image is placed, see the **-M** flag.
-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, the **mkcd** command verifies that **/mkcd/***mksysb_image* exists. If the directory does not exist, then the **mkcd** command 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 that is given with the **-V** flag, or in **rootvg** if that flag is not used. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This also requires setting the file **ulimit** size to *unlimited*.
-n	Backs up user volume group information and administration data files. This backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files, if any exist. This flag does not back up user data files. This backup can be used to create a user volume group without restoring user data files. This action cannot be done to rootvg.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.
-p *pkg_source_dir*	Names the directory or device that contains device and kernel package images. The device must 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. The **mkcd** command then prompts you to insert the writeable CD before the actual CD creation.
-P	Creates physical partition mapping during the **mksysb** or **savevg** creation. You cannot use this flag with the **-m** or **-s** flags.
-r *directory*	Indicates existing directory structure to burn onto a CD or DVD. This flag makes a CD image that is a copy of the specified directory structure.

Item	Description
-R	Prevents the **mkcd** command from removing the final CD images. **mkcd** defaults by removing everything that it creates when it finishes running. 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 by using the process ID and volume suffixes.
-s *savevg_image*	Indicates a previously created **savevg** image. All **savevg** backup images are nonbootable. See Notes for details.
-S	Stops the **mkcd** command before it writes 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 by using the process ID and volume suffixes.
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when back up is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously.
-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 the **mkcd** command restores **bosinst.data** from the specified **mksysb** image, or generates a new **bosinst.data** file during the creation of **mksysb**.
-U	Creates a **UDF** (Universal Disk Format) file system on DVD-RAM media. It does not require the amount of free space that is 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 you use the **-U** flag. You must 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.
-v *savevg_volume_group*	Denotes the volume group to be backed up using the **savevg** command. All **savevg** backup images are nonbootable. See Notes for details. For more information about where the **savevg** image is placed, see the **-M** flag.
-V *cdfs_volume_group*	Indicates the volume group that is used when you create 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 the **mkcd** command 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 completion of the command.
-w *savewpar_image*	Indicates a previously created **savewpar** image.
-W *wparname*	Denotes the workload partition to be backed up using the **savewpar** command.
-Y	Accepts licenses.
-z *custom_file*	States the full path name of the file to be copied to the root directory of the CD file system. This file could be a customization script that is 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 the **mkcd** command 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 runs.
-Z	Specifies that the Encrypted file system (EFS) information for all the files, directories, and file systems is not backed up.
-G	Excludes WPAR file systems from the system backup. The flag is not valid with **-N** flag.

Item	Description
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.

Note: Use care when you exclude file systems as a resulting backup can be unusable for system restoration.

Note:

1. If you are creating a non-bootable CD (by using the **-B** flag), you cannot use the **-p** or **-l** flags.
2. If you are creating a non-bootable CD with a **savevg** image (by using the **-s** or **-v** flags), you cannot use the **-p**, **-l**, **-u**, **-i**, **-z**, or **-b** flags.

Examples

1. To generate a bootable system backup to the CD-R device named `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1
   ```

2. To generate a system backup to the DVD-R or DVD-RAM device named `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1 -L
   ```

3. To generate a non-bootable volume group backup of the volume group `myvg` to `/dev/cd1`, enter the following command:

   ```
   mkcd -d /dev/cd1 -v myv
   ```

   Note: All **savevg** backup images are non-bootable.

4. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1**, enter the following command:

   ```
   mkcd -d /dev/cd1 -W mywpar
   ```

   Note: All **savewpar** backup images are not bootable.

5. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1** from the previously generated **savewpar** image **/wparbackups/mywpar.bff**, enter the following command:

   ```
   mkcd -d /dev/cd1 -w /wparbackups/mywpar.bff
   ```

6. To create a CD or DVD that duplicates an existing directory structure such as:

   ```
   /mycd/a
   /mycd/b/d
   /mycd/c/f/g
   ```

   enter the following command:

   ```
   mkcd -r /mycd -d /dev/cd1
   ```

   After you mount 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

Item	Description
/usr/bin/mkcd	Contains the **mkcd** command.

Related information:

backup command

savevg command

/image.data file

Creating system backups

---

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

Item	Description
-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.
-I	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

Item	Description
/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

mkcifscred Command

Purpose

Adds CIFS credentials to the **/etc/cifs_fs/cifscred** file to allow future mounting of CIFS shares with stored credentials.

Syntax

mkcifscred -h *RemoteHost* **-u** *user* [**-p** *password*]

Description

The **mkcifscred** command takes a server and user name as input, and prompts for a password. The password is encrypted, and the credentials are stored in the **cifscred** file. If the password is not passed in with the **-p** option when mounting to a CIFS server, the credentials are either retrieved from the **cifscred** file, or, if the credentials do not exist in **cifscred**, the password is prompted for and read in as hidden input.

The credentials are stored as a server/user/password set. Multiple sets of credentials for the same server are permitted with different user names. Multiple sets with the same user name on different servers are also permitted.

Flags

Item	Description
-h *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or a fully qualified domain name.
-p *password*	Specifies the password for a particular user on a particular remote host.
-u *user*	Specifies the user name whose credentials are being defined for access to the given remote host.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To add credentials for `user1` to mount on `server1`, enter:

```
mkcifscred -h server1 -u user1
```

Location

/usr/sbin/mkcifscred

Files

Item	Description
/etc/cifs_fs/cifscred	Stores the CIFS credentials.

Related information:

chcifscred command

chcifsmnt command

rmcifscred command

rmcifsmnt command

mkcifsmnt Command

Purpose

Adds a CIFS mount to the **/etc/filesystems** file and performs the mount.

Syntax

mkcifsmnt -f *MountPoint* **-d** *RemoteShare* **-h** *RemoteHost* **-c** *user* [**-p** *password*] [**-m** *MountTypeName*] [**-A** | **-a**] [**-I** | **-B** | **-N**] [**-t** {**rw** | **ro**}] [**-u** *uid*] [**-g** *gid*] [**-x** *fmode*] [**-w** *wrkgrp*]

Description

The **mkcifsmnt** command constructs a CIFS entry that is appended to the **/etc/filesystems** file. It then attempts to mount the CIFS file system. Its options are parsed and prepared to be passed into the **crfs** command, which actually adds the CIFS entry to **/etc/filesystems**.

Flags

Item	Description
-a	Specifies that the **/etc/filesystems** entry for this file system should not be automatically mounted at system restart. This is the default.
-A	Specifies that the **/etc/filesystems** entry for this file system should be automatically mounted at system restart.
-B	Specifies that the entry should be added to the **/etc/filesystems** and that it should be mounted at system restart.
-c *user*	Specifies user name used to gain access to the CIFS share.
-d *RemoteShare*	Specifies the share name on the CIFS server that should be mounted.
-f *MountPoint*	Specifies the path name over which the CIFS share should be mounted.
-g *gid*	Specifies the GID that is assigned to files in the mount. The default is 0.
-h *RemoteHost*	Specifies the name of the remote host (CIFS server). This can be provided as a host name, an IP address, or as a fully qualified domain name.
-I	Specifies that the entry should be added to the **/etc/filesystems** file, but should not be mounted.
-m *MountTypeName*	Defines the mount type that will be added to the **/etc/filesystems** file, which allows for mounting all file systems of a specific type using the **-t** option of the **mount** command. By default, no type value will be added to **/etc/filesystems**.
-N	Mounts the CIFS share with the options specified, but does not modify the **/etc/filesystems** file.
-p *password*	Specifies the password used to grant access to the specific user on the specific server. The specific credentials (server/user/password) are added to the **cifscred** file (the password will be encrypted). If the **-p** option is not specified, and the credentials do not already exist in the **cifscred** file, the command line prompts the user to provide the password, and the credentials will be added to the **cifscred** file. If the server/user credentials already exist in the **cifscred** file, this option is ignored, and the existing credentials are used for mounting.
-t {rw | ro}	Specifies whether file system should be mounted as read-only. The default is read-write (rw).
-u *uid*	Specifies the UID that is assigned to files in the mount. The default is 0.
-x *fmode*	Specifies the owner, group, and other permission bits assigned to files in the mount. The default is 755.
-w *wrkgrp*	Specifies the domain that should be used to authenticate the user during mount. If this option is not used, authentication is handled locally by the CIFS server.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

You must have root authority to run this command.

Examples

1. To add a mount over **/mnt** to `share1` on `server1`, and then authenticate as `user1`, enter:

   ```
   mkcifsmnt -f /mnt -d share1 -h server1 -c user1
   ```

Location

/usr/sbin/mkcifsmnt

Files

Item	Description
/etc/cifs_fs/cifscred	Stores the CIFS credentials.
/etc/filesystems	Stores the CIFS entry.

Related information:
chcifscred command
chcifsmnt command
rmcifscred command
rmcifsmnt command

---

mkcimreg Command

Purpose

Registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with RMC.

Syntax

To register a class:

mkcimreg [**-I** *include_directory...*] [**-f**] [**-h**] *definition_file...*

To register a provider:

mkcimreg [**-I** *include_directory...*] [**-p** *provider_directory*] [**-h**] *registration_file...*

To compile the CIM schema:

mkcimreg [**-I** *include_directory...*] **-b** *schema_path* [**-h**]

Description

The **mkcimreg** command registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with the resource monitoring and control (RMC) subsystem. You can specify one or more class definition files or provider registration files with this command. Use the **-I** flag to add directories to the search path. The output from **mkcimreg** includes the names of the files that the CIM resource manager needs for working with CIM classes.

Registering classes

If you upgrade a class using the **-f** flag (that is, if the class definition has changed somehow), you must re-register all classes that are subclasses of the upgraded class so that the changes introduced into the new class propagate to its subclasses. This must be done in "descending" order, because changes propagate from parent to child. The hierarchy is:

```
------------------
Hardware_Component
------------------
        ↑
        |
```

```
------------------
Hardware_Adapter
------------------
        ↑
        |
------------------
Hardware_Ethernet
------------------
```

If, for example, **Hardware_Component** is upgraded using **mkcimreg -f**, **Hardware_Adapter** and then **Hardware_Ethernet** must both be registered afterward, in that order.

After you register any classes:

You must restart RMC.

Restarting RMC

As the final step in the CIM class registration process, the RMC subsystem must be restarted. The sequence of commands to run follows:

1. To shut down the RMC subsystem, enter:
   ```
   /opt/rsct/bin/rmcctrl -k
   ```

   When you shut down RMC:

   Any RMC-dependent resource monitoring that is in place at the time of shutdown is deactivated. Environments that rely on RMC or any of its resource managers for high availability or other critical system functions may become temporarily disabled.
2. Wait until the following command lists the status of **ctrmc** as "inoperative":

   ```
   lssrc -s ctrmc
   ```
3. Shut down the CIM resource manager and confirm it has been stopped:
   ```
   stopsrc -s IBM.CIMRM
   lssrc -s IBM.CIMRM
   ```
4. To restart the RMC subsystem, enter:
   ```
   /opt/rsct/bin/rmcctrl -A
   ```

Registering providers

The **-p** flag indicates that the registration file on the command line contains provider registration information. The provider library's directory is expected as this flag's parameter. Provider library names follow the CMPI/Pegasus convention of appending **lib** to the beginning of the **ProviderName** property. For example, the provider with the property **ProviderName=Linux_Processor** is searched for in the **ProviderDirectory** under the name **libLinux_Processor.so**. Auxiliary libraries required by providers that are not explicitly declared in the registration file must be either in the directory supplied on the command line, or in a standard system directory such as **/usr/lib** or **/lib**.

Compiling a schema

Version 2.9 of the CIM schema is shipped with the CIM resource manager. Use the **-b** flag if you want to upgrade to a higher version. The schema file (**CIM_Schema***version*.**mof**) must be passed as the parameter to this flag. This file contains the entire CIM schema, usually in the form of a series of **#include** statements that bring in other schema MOF files.

After a CIM schema is compiled with the **-b** flag, **mkcimreg** will not need further access to the schema managed object format (MOF) files. User classes that are registered by **mkcimreg** against previous

versions of the CIM schema need to be re-registered, so changes from the new version of the schema are reflected in any derived classes.

Flags

-I *include_directory***...**
Specifies one or more additional directories to be searched.

-f Overwrites any existing class registration data with the definitions that are provided in the class definition files.

-p *provider_directory*
Specifies a path to the provider library.

-b *schema_path*
Compiles the CIM schema file.

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

Parameters

*definition_file***...**
Specifies one or more class definition files.

*registration_file***...**
Specifies one or more provider registration files.

Security

This command requires **root** authority.

Exit Status

0 The command has run successfully.

1 An internal command error occurred.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 A class registration error occurred.

Restrictions

You cannot register a class that derives from a class that has not yet been registered.

Implementation Specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard Error

When the **-T** flag is specified, this command's trace messages are written to standard error.

Examples

1. To register the **Linux_ComputerSystem** CIM class if the class definition file is located in the **$CIMDEFS** directory, enter:

   ```
   mkcimreg $CIMDEFS/Linux_ComputerSystem.mof
   ```

   You must also register the CMPI provider for this class.
2. To register a CMPI provider when the registration file is located in the **$CIMDEFS** directory and the provider library is in the **$CMPIHOME** directory, enter:

   ```
   mkcimreg -p $CMPIHOME $CIMDEFS/Linux_ComputerSystemRegistration.mof
   ```
3. To compile Version 2.12 of the CIM schema, enter:

   ```
   mkcimreg -I $SCHEMA_DIR -b CIM_Schema2.12.mof
   ```

   $SCHEMA_DIR, which indicates a search path for schema MOF files, is not required, but could help **mkcimreg** find the required MOF files if they are not in the current working directory from which the command is run.

Location

/opt/rsct/bin/mkcimreg

---

mkclass Command

Purpose

Create a Workload Management class.

Syntax

mkclass [**-a** *Attribute*=*Value ...*] [**-c** | **-m** | **-b** | **-v** | **-C** | **-B** | **-P** | **-T** | **-V** | **-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 uppercase and lowercase letters, numbers, and underscores. The name is in the format *supername* or *subname* (with the **-S** *supername* flag) or *supername.subname*. The *supername* and *subname* parameters 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 initializes the specified attribute or resource limit. See "Attributes" on page 680 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), **-A** (totalConnectTime), or **-v** (totalVirtualMemoryLimit), 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), **-L** (totalLogins), or **-V** (totalVirtualMemoryLimit) 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.

Attributes

The following attributes can be changed:

Class properties:

Item	Description
tier	Specifies the **tier** value. The **tier** value for a class is the position of the class in the hierarchy of resource limitation desirability for all classes. A class with a lower **tier** value is more favored. The **tier** value ranges from 0 through 9 (the default is 0).
inheritance	If the **inheritance** attribute is set to **yes**, the children of processes in this class remain in the class upon **exec** regardless of the automatic assignment rules in effect. If the **inheritance** attribute is set to **no**, the assignment rules apply normally. The default if not specified is **no**.
localshm	Indicates whether memory segments that are accessed by processes in different classes remain local to the class they were initially assigned to or if they go to the **Shared** class. You can specify a value of **Yes** or **No**. If not specified, the default is **No**.
authuser	Specifies the user name of the user who is allowed to assign processes to this class. The default when the attribute is not specified is **root**.
authgroup	Specifies the group name of the group of users that is allowed to assign processes to this class. There is no default value.
rset	Specifies the name of a resource set that the processes in the class have access to. By default, the class has access to all resources on the system.
vmenforce	Specifies whether all processes or only the offending processes in the class need to be terminated when the class hits the maximum VM limit. You can specify the value of **class** or **proc**. The default value is **proc**.
delshm	Specifies whether the shared segments will be deleted when the last process referencing them ends because virtual memory is exceeded. You can specify the value of **yes** or **no**. The default value is **no**.
adminuser	Specifies the user name of the user who is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default, when the attribute is not specified, is a null string, and in this case, only **root** users can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.
admingroup	Specifies the group name of the group of users that is allowed to administer the subclasses of this superclass. This attribute is valid only for superclasses. The default value, when the attribute is not specified, is a null string, meaning that no group can administer the subclasses. **Note:** If the **adminuser** or **admingroup** attribute is changed for a superclass that belongs to the running configuration (or to a configuration of the running set), a global WLM update should be performed to reflect these changes to the in-core configuration, elsewhere, updates that are restricted to superclass by such a user might fail due to lack of authority.

Class limits and shares for CPU, memory, or disk I/O resource:

Item	Description
min	Specifies the minimum percentage of the resource that must be made available when requested, expressed as a percentage of the total resource available in the system. Possible values range from 0 through 100 (the default is 0).
shares	Specifies the maximum ratio of the resource that can be made available if there is contention. This parameter is expressed in shares of the total resource available in the system. The actual ratio of the resource is dynamically computed, proportionally to the shares of all active classes. If a class has no running process, its shares are excluded from the computation. The shares are arbitrary numbers ranging from 1 through 65535. If shares is specified as a hyphen (-), the class is always considered on target and its utilization for this resource is not regulated by WLM, but the minimum and maximum limits if any still apply. This is the default if the shares for a resource are not specified.
softmax	Specifies the maximum percentage of the resource that can be made available, when there is contention. Possible values range from 1 through 100 (the default is 100). A class can exceed its soft maximum for a given resource if there is no contention on the resource.

Item	Description
hardmax	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.
max	Specifies the maximum percentage of the resource that can be made available, even if there is no contention. Possible values range from 1 through 100 (the default is 100). Specifying a value different from the default value of 100 for memory can result in some memory pages remaining unused, while some processes in the class use more.

Note: The default values for a class can be read using the **lsclass -D** command and can be changed by manually editing the property files **classes**, **shares**, or **limits** to add a default stanza. For more information about these files, see the *Files Reference*.

Class description:

Item	Description
description	The class description text can be composed of any ASCII character, except colons (**:**) and commas (**,**).

Note: This command is not supported when executed within a workload partition.

Flags

Item	Description
-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.
-v hardmax=*Value*	Specifies the virtual memory limit allowed per process in the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.

Item	Description
-V hardmax=*Value*	Specifies the virtual memory allowed for the specified class. The maximum amount of virtual memory allowed per process is (2^31)-1 for 32-bit kernels and (2^63)-1 for 64-bit kernels.

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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
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 reference:

"lsclass Command" on page 372

Related information:

wlmcntrl command

chclass command

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 System Management Interface Tool (SMIT) **smit mkclient** fast path to run this command.

Flags

Item	Description
-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

Item	Description
/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 reference:

"mkmaster Command" on page 748

Related information:

ypbind command

ypserv command

System Resource Controller

mkcluster Command

Purpose

To create a single-site cluster.

Syntax

mkcluster -r *reposdev* [**-S** *sitename* { [cle_uuid=*UUID*,cle_globid=*id*,cle_prio=*prio*] }] [**-m** *node* { [cle_ip=*addr*,cle_uuid=*UUID*,cle_globid=*id*] }] [,...] [**-d** *shareddisk* [,...]] [**-n** *clustername*] [**-s** *multi_cast_addr*] [**-c** *capability* [,...]] [-v] [**-b** backupdisk [,...]]

Description

The **mkcluster** command creates a cluster. A cluster is a collection of nodes and disks.

Each node that is added to the cluster must have common storage area network (SAN) storage devices that are zoned appropriately. The SAN storage devices are used for the cluster repository disk and for any clustered shared disks.

A multicast address is used for cluster communications between the nodes in the cluster. If any network considerations need to be reviewed before creating a cluster, consult your cluster systems administrator.

Flags

Item	Description
-b *backupdisk* [,...]	Specifies a comma-separated list of SAN shared storage device such as `hdisk5` and `hdisk6`. These disks are used as the backup for the central repository of the cluster. When the central repository is inaccessible, the disk from the list is used as a replacement. These devices must be accessible from all nodes in the site.
-c *capability* [,...]	Specifies a comma-separated list of capabilities that the cluster requires upon creation. If no capabilities are specified, the **mkcluster** command allows for the possibility that some nodes have older AIX software that is not capable of supporting newer CAA capabilities. In that case, the cluster is created in such a way that it is compatible with nodes that run older AIX software. After the cluster is created, and it is determined that all nodes can support newer CAA capabilities, the cluster automatically allows those capabilities to be used. The **-c** flag merely allows the specified capabilities to be used without first determining that all nodes are able to support it. In some situations, it is necessary for newer CAA capabilities to be enabled immediately. For example, if one or more nodes do not have IPv4 connectivity to all of the other cluster nodes, then it is necessary to specify that IPv6 capability must be enabled during cluster creation, to allow those nodes that have only IPv6 connectivity to join the cluster. All nodes must be online and able to join the cluster, to determine that the cluster can support newer capabilities. If a cluster must be created while one or more of the nodes are powered off, then specifying that the capability is needed during cluster creation allows the capability to be used before all of the nodes are able to join the cluster. However, you must make sure that all of the nodes have an AIX software level that can support that capability. Otherwise, any nodes that have older AIX software that cannot support the capability are not allowed to join the cluster. The **-c** flag supports the following capability keywords: • `ipv6`: IPv6 connectivity is required because some nodes do not have IPv4 connectivity to the rest of the cluster, or IPv6 functionality is needed before all nodes are able to join the cluster. • `site`: The cluster needs to allow one or more sites to be defined before all nodes are able to join the cluster. • `auto_repos_replace`: The created cluster can process and maintain the backup repository disks.
-c *unicast* or *multicast*	Specifies the type of communication mode that is used by CAA to transfer CAA heartbeats and other protocol messages. If the *unicast* option is specified, CAA uses unicasting to transfer the protocol messages. If the *multicast* option is specified, CAA uses multicasting to transfer the protocol messages. If no option is specified, CAA uses the default multicast communication mode.
-r *reposdev*	Specifies the name of the SAN shared storage device that is used as the central repository for the cluster configuration data, such as *hdisk10*. This device must be accessible from all gateway nodes in the site. It is required that this device is a minimum of 1 GB, and is backed up by a redundant and highly available SAN configuration.
-S *sitename*	Specifies the name of the local site. If not specified, a default site with the name LOCAL is created. Currently, a cluster can support only 2 sites. To create a second site, use the **chcluster** command. The following site information can be specified: • `cle_uuid`: The site UUID, which is acknowledged as unique across the cluster. If not specified, the site UUID is automatically generated. • `cle_globid`: The short ID of site, which must be a unique unsigned number greater than zero. If not specified, the site short ID is automatically generated. The following site attribute can be specified: • `cle_prio`: The priority of a site. A lower value indicates a higher priority. The priority is used in the context of synchronizing the repository metadata. If two sites split and the repository data becomes out of sync, then the data from the site with higher priority must be copied over to the site with lower priority.

Item	Description
-m *node*[,...]	Lists the comma-separated host names or IP addresses for nodes that are members of the cluster. The local host must be included in the list. If the **-m** flag is not used, the local host is implied, causing a one-node local cluster to be created. The following node information can be specified: • `cle_uuid`: The node UUID, which is acknowledged as unique across the cluster. If not specified, the node UUID is automatically generated. • `cle_globid`: The short ID of node, which must be a unique unsigned number greater than zero. If not specified, the node short ID is automatically generated. The following node attributes can be specified: • `cle_ip`: The nodes gateway address (in case the cluster spans across multiple sites). Typically, this attribute is an address through which this node can be reached from an external node. This address can be specified in either IP version 4 or version 6 format.
-d *shareddisk*[,...]	Specifies a comma-separated list of shared storage area network (SAN) devices, such as *hdisk12, hdisk34,* to be incorporated into the cluster configuration. Specified devices must not be open when the **mkcluster** command is run.
-n *clustername*	Sets the name of the cluster that is being created. If no name is specified when you run the **mkcluster** command, a default of **CL_hostname** is used, where *hostname* is the name of the local host. You can retrieve the name of the local host by running the **gethostname()** function.
-s *multi_cast_addr*	Sets the multicast address of the cluster that is created. This address is used for internal communication within the cluster. Only a multicast address in IPv4 format is accepted. If an IPv6 multicast address is needed, it must be generated from the IPv4 address. If the **-s** flag is not specified when you first run the **mkcluster** command, the necessary multicast addresses are automatically generated.
-v	Specifies the verbose mode.

Examples

1. To create a cluster of one node and use the default values, enter the following command:
   ```
   mkcluster -r hdisk
   ```

   The output is a cluster that is name CL_*myhostname* with a single node in the cluster. The multicast address is automatically generated and no shared disks are created for this cluster. The
   ```
   mkcluster -r hdisk1
   ```

   repository is set up on *hdisk1* and this disk cannot be used by the node for any other purpose. The repository device is dedicated as the cluster repository disk.
2. To create a multinode cluster, enter the following command:
   ```
   mkcluster -n mycluster -m nodeA,nodeB,nodeC
   ```

   The output is a cluster of three nodes and uses the default values. The output also creates a cluster with the specified name and the multicast address is automatically created. Three disks are created as shared clustered disks for this cluster. The repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being the cluster repository disk. A volume group of cvg is created for the cluster repository disk and these logical volumes are used exclusively by the clustering subsystem.
3. To create a cluster that is capable of IPv6 and sites, enter the following command:
   ```
   mkcluster -n mycluster -m nodeA,nodeB,nodeC -r hdisk1 -c ipv6, site
   ```

   This command creates a cluster of three nodes that are immediately capable of using IPv6 networks and having sites that are defined. The cluster is named *mycluster*, and the local site and multicast address are automatically created with default values. If any IPv6 networks are configured, they can be used for cluster communication. The repository device is set up on hdisk1, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.
4. To create a cluster with one site named *mysite*, enter the following command:

```
mkcluster -n mycluster -S mysite -m nodeA,nodeB,nodeC -r hdisk1 -d hdisk10,hdisk11,hdisk12
```

The output is a single-site cluster of three nodes that uses the default attribute values for all sites and all nodes. The cluster is named *mycluster*, and the local site is named *mysite*. The multicast address is automatically created. Three disks are created as shared clustered disks for the local site. The repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.

5. To create a cluster with one site named *mysite*, specifying site and node information, enter the following command:

```
mkcluster -n mycluster -S mysite{cle_uuid=0551c722-92fe-11e1-97b0-1aae1ed14715,
cle_globid=5,cle_prio=2}
-m nodeA,nodeB,nodeC{cle_uuid=e4ad47bc-92fd-11e1-8486-1aae1ed14715}
-r hdisk1 -d hdisk10,hdisk11,hdisk12
```

The output is a single-site cluster of three nodes. Nodes nodeA and nodeB have automatically generated UUIDs, while nodeC has a UUID of e4ad47bc-92fd-11e1-8486-1aae1ed14715. The cluster is named *mycluster*, and the local site is named *mysite* and has a UUID of 0551c722-92fe-11e1-97b0-1aae1ed14715, a short ID of 5, and a priority of 2. The multicast address is automatically created. Three disks are created as shared clustered disks for the local site. The repository device is set up on *hdisk1*, and it cannot be used by any of the nodes for any other purpose. The repository device is now dedicated to being a cluster repository disk.

6. To create a multinode unicast cluster with one site named *mycluster*, append **-c** *unicast* to the **mkcluster** command. The multicast cluster in example 2 can be made unicast by entering the following command:

```
mkcluster -r hdisk10 -m nodeA,nodeB,nodeC -n mycluster -r hdisk1
-d hdisk10,hdisk11,hdisk12 -c unicast
```

7. To create a cluster that uses backup disks that are added later, enter the following command:

```
mkcluster -n mycluster -n nodeA,nodeB -r hdisk1 -c auto_replace_repost
```

8. To create a cluster and populate the backup repository disk list, enter the following command:

```
mkcluster -n mycluster -n nodeA,nodeB -r hdisk1 -b hdisk5,hdisk6
```

mkcomg Command

Purpose

Creates a new communication group definition for a peer domain.

Syntax

mkcomg [**-s** *sensitivity*] [**-p** *period*] [**-g** *grace*] [**-t** *priority*] [**-x b** | **r** | **br**] [**-N** *UseForNodeMembership*] [**-e** *NIM_path*] [**-m** *NIM_parameters*] [**-M** *media_type*] [**-i** {**h** | **n**}**:***interface1*[**:***node1*][**,***interface2*[**:***node2*]...] | **-S** {**h** | **n**}**:"***interface_selection_string***"** [**-6**] [**-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 amount of time between heartbeats. The period is specified in seconds and is significant to milliseconds. It can be specified as an integer or as a floating-point number.

-g *grace*
: Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

  The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

  0 Specifies that the grace period is disabled.

  -1 | D Specifies that the topology services subsystem controls the grace period. This is the default.

-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 b | r | 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**.

-N *UseForNodeMembership*
: Specifies whether group services will use the communication group in calculating node membership. Sets the **UseForNodeMembership** persistent resource attribute for the communication group resource. Valid values are:

  0 Indicates that, regardless of the results of liveness checks run on **NetworkInterface**

resources that are members of this communication group, group services will not use those results in calculating whether the node owning the interfaces is online.

1 Indicates that group services will use the results of liveness checks run on the **NetworkInterface** resources in calculating the online state of their owning nodes.

-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.

-M *media_type*
Specifies the type of interfaces that make up *communication_group*. Valid values are:

0 Indicates that *communication_group* consists of interface resources other than IP or disk.

1 Indicates that *communication_group* consists of IPv4 or IPv6 interface resources.

If the **-M** flag is not specified, this is the default.

2 Indicates that *communication_group* consists of disk interface resources.

-i {h | n}:*interface1***[:***node1***] [,***interface2***[:***node2***]...**
Assigns *communication_group* to one or more heartbeat or network interface resources and, optionally, to the nodes where these resources can be found. Specify **-i h** for heartbeat interface resources or **-i n** for network interface resources.

By default, the **-i n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-i n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

If **-i** is specified, **-S** cannot be specified.

-S {h | n}:"*network_selection_string***"**
Assigns *communication_group* to the heartbeat or network interface that is specified by *interface_selection_string*. Specify **-S h** for heartbeat interfaces or **-S n** for network interfaces.

By default, the **-S n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-S n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

If **-S** is specified, **-i** cannot be specified.

-6 Specifies that IPv6 addresses represented as resources on each interface have their communication group changed to the one specified. IPv4 addresses represented as resources on the interfaces would be unaffected.

By default (without **-6** specified), the inverse is true. Only IPv4 addresses represented as resources on the interface would have their communication group changed.

-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.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

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 **/opt/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 /opt/rsct/bin/hats_nim -m "-l 5" ComGrp1
   ```

6. To define the communication group **ComGrp1** for **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource named **hbi0** on **nodeC**, run this command on **nodeA**:

   ```
   mkcomg -i h:hbi0:nodeC ComGrp1
   ```

7. 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
   ```

8. To define the communication group **ComGrp1** for **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource that uses the subnet 9.345.67.812, run this command on **nodeA**:

   ```
   mkcomg -S h:"Subnet == 9.345.67.812" ComGrp1
   ```

9. 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
   ```

10. To define the communication group **ComGrp1** for **ApplDomain**, using a period of 500 milliseconds, run this command on **nodeA**:

    ```
    mkcomg -p 0.5 ComGrp1
    ```

Location

/opt/rsct/bin/mkcomg

---

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***"] [-b** *interval*[**,**_max_events_][**,**_retention_period_][**,**_max_totalsize_]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,**_node_name2..._]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s "***selection_string***"] [-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-h**] [**-TV**] *condition*

mkcondition -c *existing_condition*[**:**_node_name_] [**-r** *resource_class*] [**-e "***event_expression***"] [-E "***rearm_expression***"] [-d "***event_description***"] [-D "***rearm_description***"] [-b** *interval*[**,**_max_events_][**,**_retention_period_][**,**_max_totalsize_]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,**_node_name2..._]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s "***selection_string***"] [-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-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). Use the **mkresponse** command to define one or more responses to an event. You can then link the conditions to the responses using the **mkcondresp** command, or you can use the **startcondresp** command to link the responses and start monitoring.

Using the **-b** flag, multiple events can be batched or grouped together and passed to a response. The grouping of events is by the time span in which they occur. In addition, the grouping can be done such that a specified maximum number of events are grouped within the time span. A response that handles batched events must be defined as supporting batched events.

In a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the condition definition. If you are using **mkcondition** on the management server and you want the condition to be defined on the management server, do *not* specify the **-p** flag. 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.

To lock a condition so it cannot be modified or removed, use the **chcondition** command (with its **-L** flag).

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-b *interval*[**,**_max_events_][**,**_retention_period_][**,**_max_totalsize_]
: Specifies one or more batching-related attributes. Use commas to separate the attribute values. Do not insert any spaces between the values or the commas.

interval specifies that the events are to be batched together for the indicated interval. Batching continues until no events are generated for an interval. Use an interval of 0 to turn batching off.

max_events specifies that the events are to be batched together until the *max_events* number of events are generated. The interval restarts if the *max_events* number of events is reached before the interval expires.

retention_period specifies the retention period in hours. The batched event file is saved for the time specified as the retention period. Once this time is reached, the file is automatically deleted.

max_totalsize specifies the total size for the batched event file in megabytes (MB). The batched event file is saved until this size is reached, Once the size is reached, the file is automatically deleted.

max_events, *retention_period*, and *max_totalsize* cannot be specified unless interval is greater than 0.

When *interval* is greater than 0 and *max_events* is 0, no maximum number of events is used.

If *retention_period* and *max_totalsize* are both specified, the batched event file is saved until the specified time or size is reached, whichever occurs first.

If you want to change one, two, or three attribute values, you must specify a valid value or an empty field for any attributes that precede the value you want to change. You do not have to specify any values for attributes that follow the value you want to change. For example, if you only need to change the retention period, you need to specify values for interval and *max_events* as well. You can provide an empty field if an attribute does not need to be changed. To change the retention period to 36 hours without changing the values of interval and *max_events*, enter:

```
mkcondition -c existing_condition -b ,,36
```

-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.

-d "*event_description***"**
Describes the event expression.

-D "*rearm_description***"**
Describes the rearm expression.

-e "*event_expression***"**
Specifies an *event expression,* which determines when an event occurs. An event expression consists of a dynamic attribute or a persistent attribute of *resource_class*, a mathematical comparison symbol (or **<**, for example), and a constant. When this expression evaluates to TRUE, an event is generated.

-E "*rearm_expression***"**
Specifies a rearm expression. After *event_expression* has evaluated to True and an event is generated, the rearm expression determines when monitoring for the event expression will begin again. Typically, the rearm expression prevents multiple events from being generated for the same event evaluation. The rearm expression consists of dynamic attributes or persistent attributes of *resource_class*, mathematical comparison symbols (> or <, for example), logical operators (|| or &&), constants, and an optional qualifier.

--g 0 | 1 | 2
Specifies granularity levels that control audit logging for the condition. The levels of granularity are:

0 Enables audit logging. ERRM writes all activities to the audit log. This is the default.

1 Enables error logging only. ERRM writes only in case of errors to the audit log.

2 Disables audit logging. ERRM does not write any records to the audit log.

-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.

-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. Node group names can also be specified, which are expanded into a list of node names.

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.

The host name does not have to be online in the current configuration, but once the condition is monitored, the condition will be in error if the node does not exist. The condition will remain in error until the node is valid.

-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.

If you are using **mkcondition** on the management server and you want the condition to be defined on the management server, do *not* specify the **-p** flag.

--qnotoggle
Specifies that monitoring does not toggle between the event expression and the rearm expression, but instead the event expression is always evaluated.

--qtoggle
Specifies that monitoring toggles between the event expression and the rearm expression.

-r *resource_class*
Specifies the resource class to be monitored by this condition. You can display the resource class names using the **lsrsrcdef** command.

-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 (the default)

-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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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"
```

5. To define a condition with the name "vmstat is running" to monitor when user **joe** is running the **vmstat** program in a 64-bit environment, enter:

```
mkcondition -r "IBM.Program" \
-e "Processes.CurPidCount > 0" -E "Processes.CurPidCount  <= 0"  \
-d "Generate event when user starts vmstat" \
-D "Restart monitoring when vmstat is terminated" \
-s ProgramName == \"vmstat64\" && Filter==\"ruser==\\\"joe\\\"\"" \
-S "i" -m "l" "vmstat is running"
```

6. To define a condition with the name "myscript terminated" to monitor when a script has ended, enter:

```
mkcondition -r "IBM.Program" \
-e "Processes.CurPidCount <= 0" -E "Processes.CurPidCount  > 0"  \
-d "Generate event when myscript is down" \
-D "Rearm the event when myscript is running" \
-s ProgramName == \"ksh\" && Filter == 'args[1]==\"/home/joe/myscript\"'" \
-m "l" "myscript terminated"
```

In this example, **args** represents the array of argument strings that was passed to **main**. Because this is an array, **args[1]** references the first argument after the program name. Use the **ps -el** command to determine the **ProgramName**. See the **lsrsrcdef** command for more information.

7. To batch together a maximum of 20 events at a time that come from a sensor named **DBInit** in 60-second intervals, enter:

```
mkcondition -r "IBM.Sensor" \
-e "Int32 < 0" -E "Int32 > 0"  -b 60,20 \
-s "Name == \"DBInit\""  "DBInit Sensor"
```

8. To define a condition with the name tmp space used to check for percentage of space used greater than 90% for **/tmp** for at least seven out of the last 10 observations, including comments, enter:

```
mkcondition -r IBM.FileSystem \
-e "PercentTotUsed > 90 __QUAL_COUNT(7,10)" \
-d "Generate event when tmp > 90% full for 7 out of 10 last
\observations" \ -s 'Name=="/tmp"' "tmp space used"
```

9. To define a condition with the name adapter stability to check for adapter status that has changed four times within one minute, including comments, enter:

```
mkcondition -r IBM.NetworkInterface \
-e "OpState != OpState@P __QUAL_RATE(4,60)" \
-d "Generate event when OpState is changed 4 times within 1 minute" \
"adapter stability"
```

10. To define a condition for a batched event called tmp space used to check the percentage of space used by **/tmp** that is greater then 90%, with a batch interval of 5 and a batch event file retention period of 72 hours, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -b 5,,72 "tmp space used"
```

11. To define a condition called tmp space used to check that percentage of space used by **/tmp** that is greater then 90%, with audit logging enabled only in case of errors, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -g 1 "tmp 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" -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 p \
"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" -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 -p nodeB "local FileSystem space used"
```

Location

/opt/rsct/bin/mkcondition

---

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. A link between a condition and a response is called a *condition/response association.* This command creates one or more condition/response associations; it 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 using the **startcondresp** command.

To lock a condition/response association, use the **-L** flag of the **rmcondresp**, **startcondresp**, or **stopcondresp** 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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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 anytime", run this command:

   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

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 management server 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

/opt/rsct/bin/mkcondresp

---

mkcosi Command

Purpose

Makes a Common Operating System Image (COSI) for use with thin servers.

Syntax

mkcosi -s *Source* [**-l** *Location*] [**-S** *Server*] [**-v**] *COSI*

Description

The **mkcosi** command creates a Common Operating System Image (COSI). A COSI is a repository that contains all the necessary software to bring a thin server up to a functional state. The **mkcosi** command takes a source (**-s** *Source*) containing installable images and attempts to install those software images into a specific location (**-l** *Location*). If the **-S** *Server* is specified, the COSI image is stored on that particular server. The result is an OS image that can be used by thin servers as its boot image and operating system.

This command is dependent upon the **bos.sysmgt.nim.master** fileset being present on the system. When this command is executed for the first time, the machine executing the command is configured as a NIM master. The **mkcosi** command uses the **nim_master_setup** command to configure the machine as a NIM master. The **-S** parameter must point to a machine that is managed by the caller of the **mkcosi** command.

Flags

Item	Description
-l	Specifies the full path name to a location for storing the COSI.
-S *Server*	Specifies the name of the machine where the COSI image resides.
-s *Source*	Specifies the source of installable images to be used in creating the COSI. The source can be an **lpp_source**, a device with installable media, a directory to installable images, or a remote location to installable images.
-v	Enables verbose debug output when the **mkcosi** command runs.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **mkcosi** command.

Examples

1. To define a COSI named `cosi1` from a CD-ROM `cd0`, and to store it at the location `/export/cosi1`, enter the following command:

   ```
   mkcosi -s cd0 -l /export/cosi1 cosi1
   ```

Location

/usr/sbin/mkcosi

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

Related information:

cpcosi command

nim command

nimconfig command

rmcosi command

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 it is complete could result in a corrupted database.

The **mkdev** command performs the following:

- 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)
- 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 you need 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 previously-defined parents of the specified device that are not already configured. 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 of the device. 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 in the specified **-f** *File* flag.

You can use the System Management Interface Tool (SMIT) **smit mkdev** fast path to run this command.

Flags

Item	Description
-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. Multiple attribute-value 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 necessary flags from the *File* parameter.
-h	Displays the command usage message.
-l *Name*	Specifies the predefined 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 it is 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 cannot be used 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.

Exit Status

This command returns the following exit values:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Privilege Control: Only the root user and members of the system group should have execute (x) access to this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 connected to the `scsi0` SCSI adapter and using SCSI ID 5 and LUN of 0, type the following:

   ```
   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 predefined `rmt0` tape device available to use, type the following:

   ```
   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 the following:

   ```
   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

Item	Description
/usr/sbin/mkdev	Contains the **mkdev** command.

Related reference:

"lsattr Command" on page 354

"lsconn Command" on page 390

"lsdev Command" on page 396

"lsparent Command" on page 472

mkdir Command

Purpose

Creates one or more new directories.

Syntax

mkdir [**-e**] [**-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

Item	Description
-e	Creates directories with encryption inheritance.
-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:

Item	Description
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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/mkdir	Contains the **mkdir** command.

Related information:

chmod command

mkdir command

umask command

File and directory access modes

---

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 reference:

"mkdir Command" on page 703

---

mkdom Command

Purpose

Creates a new domain.

Syntax

mkdom [**-R** *load_module*] [*Attribute* **=** *Value* ...] *Name*

Description

The **mkdom** command creates a new domain in the domain database. The domain attributes can be set during the domain creation phase by using the *Attribute* **=** *Value* parameter.

When the system is operating in the enhanced Role Based Access Control (RBAC) mode, modifications made to the domain database are not used for security considerations until the database is sent to the kernel security tables by using the **setkst** command.

Note: The domain id value can be lesser than or equal to 1024. The **mkdom** command enables you to create 1024 domains on the system.

If the system is configured to use multiple authentication load modules for the role-based access control (RBAC) domain database, the new RBAC domain is created in the first load module specified by the `secorder` attribute in the `domains` stanza of the `/etc/nscontrol.conf` file. Use the **-R** flag to create an RBAC domain in a specific authentication load module.

Flags

Item	Description
-R *load_module*	Specifies the loadable module that is to be used when you create an RBAC domain.

Parameters

Item	Description
Attribute **=** *Value*	Initializes a domain attribute. See the **chdom** command for valid attributes and values.
Name	Specifies a unique domain name string.

Restrictions on creating domain names: The *Name* parameter specified must be unique and is limited to a 63 single-byte printable character. While the **mkdom** command supports multibyte domain names, it is recommended that you restrict domain names to characters within the POSIX portable file name character set. Domain names must not begin with a - (dash), + (plus sign), @ (at sign), or ~ (tilde) and must not contain any space, tab, or new-line characters. You cannot use the keywords ALL, default, ALLOW_OWNER, ALLOW_GROUP, ALLOW_ALL or * as a domain name. Additionally, do not use any of the following characters within a domain string:

Item	Description
:	Colon
"	Double quotation mark
#	Number sign
,	Comma
=	Equal sign
\	Backslash
/	Forward slash
?	Question mark
'	Single quotation marks
`	Grave accent

Security

The **mkdom** command is a privileged command. Callers of the command must have activated a role that has the following authorization to run the command successfully.

Item	Description
aix.security.domains.create	Required to run the command.

Files Accessed

Item	Description
File	Mode
/etc/security/domains	rw

Examples

1. To create a domain `hrdom` and to have the **mkdom** command assign an appropriate ID value, enter the following command:

   ```
   mkdom hrdom
   ```

2. To create a custom domain in Lightweight Directory Access Protocol (LDAP), enter the following command:

   ```
   mkdom -R LDAP custom
   ```

Related information:

lsdom command

setkst command

getdomattr command

/etc/security/domains file

---

mkdvd Command

Purpose

Creates multi-volume DVDs from a **mksysb**, **savevg**, or **savewpar** backup image.

Syntax

mkdvd -r *directory* | **-d** *dvddevice* | **-S [-m** *mksysbimage* | **-M** *mksysbtarget* | **-s** *savevgimage* | **-v** *savevgvolumegroup* | **-w** *savewpar_image* | **-W** *wparname*] [**-C** *cdfsdir*] [**-I** *cdimagedir*] [**-V** *dvdfsvolumegroup*] [**-B**] [**-p** *pkgsourcedir*] [**-R** | **-S**] [**-i** *image.data*] [**-u** *bosinst.data*] [**-f** *wparspecificationfile*] [**-e**] [**-P**] [**-l** *packagelist*] [**-b** *bundlefile*] [**-z** *customfile*] [**-D**] [**-U**] [**-Y**] [**-n**] [**-a**] [**-A**] [**-c**] [-Z] [**-G** | **-N**] [**-x** *file*] [**-T**]

Description

The **mkdvd** command creates a system backup image (**mksysb**) to DVD-Recordable (DVD-R, DVD-RAM) from the system **rootvg** or from a previously created **mksysb** image. It creates a volume group backup image (**savevg**) to DVD from a user-specified volume group or from a previously created **savevg** image. It also creates the backup image of a workload partition (**savewpar**) to DVD from a user-specified workload partition or from a previously created **savewpar** image.

Note: If the system has a **multibos** environment where both instances are mounted, you can restore the backup only by using the **alt_disk_mksysb** command.

For DVD media, system backups that are made with the **mkdvd** command have a limitation in that they expect the media to be 4.7 GB or larger per side. The **mkdvd** command does 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 you go beyond the media capacity.

When a bootable backup of a root volume group is created, the boot image reflects the currently running kernel. If the current kernel is the 64-bit kernel, the backup boot image is also 64 bit, and it boots 64-bit systems only. If the current kernel is a 32-bit kernel, the backup boot image is 32 bit, and it can boot both 32-bit and 64-bit systems.

With the **mkdvd** command, you can create bootable and non-bootable DVDs in Rock Ridge (ISO9660) or UDF (Universal Disk Format) format.

Note: The functionality that is required to create Rock Ridge format DVD images and to write the DVD image to the DVD-RAM device is not part of the **mkdvd** command. You must supply additional code to the **mkdvd** command to do these tasks. You can call the code by using shell scripts and then link it to **/usr/sbin/mkrr_fs** (for creating the Rock Ridge format image) and **/usr/sbin/burn_cd** (for writing to the DVD device). Both links are called from the **mkdvd** 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, the **mkdvd** command creates the necessary file systems and removes them when the command finishes running. File systems that you supply are checked for adequate space and write access.

Note: If the **mkdvd** command creates file systems in the backup volume group, they are excluded from the backup.

If you must create multi-volume DVDs because the volume group image does not fit on one DVD, the **mkdvd** command provides instructions for DVD replacement and removal until all the volumes have been created.

Flags

Item	Description
-a	Does not back up extended attributes or NFS4 ACLs.
-A	Backs up DMAPI file system files.
-b *bundlefile*	Gives the full path name of the file that contains 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 DVD file system and also copied to RAM in case the DVD is unmounted. The file would be listed as BUNDLES=**/../usr/sys/inst.data/user_bundles/***bundlefile* in the **bosinst.data** file.
-B	Prevents **mkdvd** from adding boot images (non-bootable DVD) to the DVD. Use this flag if you create a **mksysb** DVD that you will not boot. Before you install the non-bootable **mksysb** DVD, you must boot a same level (V.R.M.) product media. The **mkdvd** command defaults to creating a bootable DVD for the machine type of the source system. For more information, see the Notes section.
-c	Does not compress or pack files as they are backed up.

Item	Description
-C *cdfsdir*	Specifies the file system that is used to create the DVD file system structure, which must have up to 4.38 GB for DVD sized images. The DVD image consumes only as much room as is necessary to contain all the data on the DVD. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory exists, the **mkdvd** command uses that directory. If you do not specify the **-C** flag and the **/mkcd/cd_fs** directory does not exist, the **mkdvd** command creates the file system **/mkcd/cd_fs** and removes it when the command finishes running. The command creates the file system in the volume group that is indicated with the **-V** flag, or **rootvg** if that flag is not used. Each time that you invoke the **mkdvd** command, a unique subdirectory (by using the process id) is created under the **/mkcd/cd_fs** directory, or in the directory that is specified with the **-C** flag. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-d *dvddevice*	Indicates the DVD-R or DVD-RAM device (`/dev/cd1`, for instance). This flag is required unless you use the **-S** flag.
-D	Turns on the debug output information feature. The default is no debug output.
-e	Excludes the files and directories from the backup image that is listed in the **/etc/exclude.***volume_group* file. You cannot use this flag with the **-m** or **-s** flags.
-f *wparspecificationfile*	Specifies the user-supplied WPAR specification file. This specification file of workload partition takes precedence over the **wpar.spec** file in the **savewpar** image. If you do not use the **-f** flag, the **mkdvd** command restores the **wpar.spec** from the specified **savewpar** image, or generates a new **wpar.spec** file during the creation of **savewpar**.
-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 specify the **-i** flag, then the **mkdvd** command 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.
-I *cdimagesdir*	Specifies the directory or file system where the final DVD images are stored before they are written to the DVD-R or DVD-RAM device. If this flag is not used, the **mkdvd** command 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 that is given with the **-V** flag, or in **rootvg** if that flag is not used. If the **mkdvd** command 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 you create multi-volume DVDs. The DVD image consumes only as much room as is necessary to contain all the data on the DVD. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-l *packagelist*	Specifies the file that contains a list of additional packages you want copied to the **./usr/lpp/inst.images** directory of the DVD file system. The images are copied from the location that is named with the **-p** flag. If you use the **-l** flag, you must also use the **-p** flag.
-m *mksysbimage*	Specifies a previously created **mksysb** image. If you do not specify the **-m** flag, the **mkdvd** command calls **mksysb**. For more information about where the **mksysb** image is placed, see the **-M** flag.
-M *mksysbtarget*	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, the **mkdvd** command verifies that **/mkcd/***mksysbimage* exists. If the directory does not exist, then the **mkdvd** command creates a separate file system, **/mkcd/***mksysbimage*, where the **mksysb** or **savevg** images are temporarily stored. The command creates the file system in the volume group that is given with the **-V** flag, or in **rootvg** if that flag is not used. **Note:** If performing DVD sized backups, the file systems must be *large file enabled*. This type of backup also requires setting the file **ulimit** size to *unlimited*.
-n	Backs up user volume group information and administration data files. This flag backs up files such as **/tmp/vgdata/vgname/vgname.data** and map files, if any exist. This flag does not back up user data files. This backup can be used to create a user volume group without restoring user data files. This action cannot be done to rootvg.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.

Item	Description
-p *pkgsourcedir*	Names the directory or device that contains device and kernel package images. The device must be a CD or DVD device (for example, **/dev/cd0**). If you use the same DVD-R or DVD-RAM device that you gave with the **-d** flag, the product media must be inserted into the drive first. The **mkdvd** command then prompts you to insert the writable DVD before the actual DVD creation.
-P	Creates physical partition mapping during the **mksysb** or **savevg** creation. You cannot use this flag with the **-m** or **-s** flags.
-r *directory*	Indicates existing directory structure to burn onto a DVD. This flag makes a DVD image that is a copy of the specified directory structure.
-R	Prevents the **mkdvd** command from removing the final DVD images. The **mkdvd** command defaults by removing everything that it creates when it finishes running. The **-R** flag allows multiple DVD image sets to be stored, or for DVD creation (burn) to occur on another system. If multiple volumes are needed, the final images are uniquely named by using the process ID and volume suffixes.
-s *savevgimage*	Indicates a previously created **savevg** image. See Notes for details.
-S	Stops the **mkdvd** command before it writes to the DVD-R or DVD-RAM without removing the final DVD images. The **-S** flag allows multiple DVD sets to be created, or for DVDs 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 by using the process ID and volume suffixes.
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when back up is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously.
-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 specify the **-u** flag, then the **mkdvd** command restores **bosinst.data** from the specified **mksysb** image, or generates a new **bosinst.data** file during the creation of **mksysb**.
-U	Creates a **UDF** (Universal Disk Format) file system on DVD-RAM media. It does not require the amount of free space that is 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 you use the **-U** flag. You must 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.
-v *savevgvolumegroup*	Denotes the volume group to be backed up using the **savevg** command. See Notes for details. For more information about where the **savevg** image is placed, see the **-M** flag.
-V *dvdfsvolumegroup*	Indicates the volume group that is used when you create the file systems needed for the **mkdvd** 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 the **mkdvd** command creates the file systems in the backup volume group, those file systems are not included as part of the backup image. The **mkdvd**-created file systems are removed upon the command's completion.
-w *savewparimage*	Indicates a previously created **savewpar** image.
-W *wparname*	Denotes the workload partition to be backed up using the **savewpar** command.
-Y	Accepts licenses.
-z *customsfile*	States the full path name of the file to be copied to the root directory of the DVD file system. This file can be a customization script that is 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 **mkdvd** 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 runs.
-Z	Specifies that the Encrypted file system (EFS) information for all the files, directories, and file systems is not backed up.
-G	Excludes WPAR file systems from the system backup. This flag is not valid with **-N** flag.

Item	Description
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.

Note: Use care when you exclude file systems as a resulting backup can be unusable for system restoration.

Note:

- If you are creating a non-bootable DVD (by using the **-B** flag), you cannot use the **-p** or **-l** flags.
- If you are creating a non-bootable DVD with a **savevg** image (by using the **-s** or **-v** flags), you cannot use the **-p**, **-l**, **-u**, **-i**, **-z**, or **-b** flags.

Examples

1. To generate a bootable system backup to the DVD-R device named `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1
   ```

2. To generate a system backup to the DVD-R or DVD-RAM device named `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1
   ```

3. To generate a non-bootable volume group backup of the volume group `myvg` to `/dev/cd1`, enter the following command:

   ```
   mkdvd -d /dev/cd1 -v myvg
   ```

   Note: All **savevg** backup images are non-bootable.

4. To generate a non-bootable backup of the workload partition `mywpar` to /dev/cd1, enter the following command:

   ```
   mkdvd -d /dev/cd1 -W mywpar
   ```

   Note: All **savewpar** backup images are not bootable.

5. To generate a non-bootable backup of the workload partition `mywpar` to **/dev/cd1** from the previously generated **savewpar** image **/wparbackups/mywpar.bff**, enter the following command:

   ```
   mkdvd -d /dev/cd1 -w /wparbackups/mywpar.bff
   ```

6. To create a DVD or DVD that duplicates an existing directory structure such as the following example:

   ```
   /mycd/a
   /mycd/b/d
   /mycd/c/f/g
   ```

   enter the following command:

   ```
   mkdvd -r /mycd -d /dev/cd1
   ```

   After you mount 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

Item	Description
/usr/bin/mkdvd	Contains the **mkdvd** command.

Related information:
backup command
savewpar command
/image.data file
Creating system backups

---

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

Item	Description
-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:

Item	Description
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

Item	Description
/usr/bin/mkfifo	Contains the **mkfifo** command.

Related information:

chmod subroutine

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. IPsec filter rules for this command can be configured using the **genfilt** command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 System Management Interface Tool (SMIT) **smit mkfont** fast path to run this command.

Parameter

Item	Description
FontPathName	The fully qualified pathname of a font file.

Security

The user must have root authority to execute this command.

Example

To add the font file **/usr/lpp/fonts/Rom10.snf**, enter the following command:

```
mkfont   /usr/lpp/fonts/Rom10.snf
```

Files

Item	Description
/bin/mkfont	Contains the **mkfont** command.
/usr/lpp/fonts	Contains the font directory.

Related reference:

"lsfont Command" on page 410

Related information:

chfont command

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:

Item	Description
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

Item	Description
/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:

Item	Description
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*

Note:

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 file system on a striped logical volume, refer to "File Systems on Striped Logical Volumes" on page 741 the **mklv** documentation.
4. To create a JFS2 file system on a logical volume, the minor number of the logical volume must be greater than 3071.

Flags

Item	Description
-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):

Item	Description
-o ag={ **8** | **16** | **32** | **64** }	Specifies the allocation group size in megabytes. An allocation group is a grouping of i-nodes 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 JFS and large files 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.

Notes:

- 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.
- The root file system (**/**) cannot be compressed.
- 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:

Item	Description
-o agblksize={ 512 | 1024 | 2048 | 4096 }	Specifies the Enhanced Journaled File System (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 isnapshot={yes|no}	Specifies whether the file system can support internal snapshots. Specifying **yes** enables the file system to support internal snapshots and **v2** extended attributes. The resulting file system is not compatible with releases earlier than AIX 6.1.
-o name=*mountpoint*	Specifies the mount point for the file system.
-o log=*LVName*	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 2047 MBytes and cannot be greater than 10% of the size of the file system.
-o ea={v1 | v2}	Specifies the format to be used to store named extended attributes in the JFS2 file system. The **v2** format provides support for scalable named extended attributes as well as support for NFS4 ACLs. The **v1** format is compatible with prior releases of AIX. The default format is **v1**.
-o efs={yes|no}	Specifies encryption. Specifying **yes** enables encryption for the JFS2 file system. • If the **efs** attribute is set to **yes**, the **mkfs** command automatically creates the JFS2 file system with the extended attribute format set to v2. The **ea** attribute is not required. • If the **efs** attribute is set to **no**, the **mkfs** command creates a file system that is not encrypted.

Item	Description
-o vix={yes|no}	Specifies whether the file system can allocate i-node extents smaller than the default of 16 KB, if there are no contiguous 16 KB extents free in the file system. After a file system is enabled for small free extents, the file system cannot be accessed on AIX 5.1 or earlier releases. **yes** The file system can allocate variable-length i-node extents. This is the default value beginning with AIX 6.1. **no** The file system must use the default size of 16 KB for i-node extents. This has no effect if the file system already contains variable-length i-node extents.
-o maxext=Value	Specifies the maximum size of a file extent in file system blocks. A zero value implies that the JFS2 default maximum should be used. Values less than 0 or exceeding maximum supported extent size of 16777215 are invalid.

Note: The **agblksize** attribute is set at file system creation and cannot be changed after the file system is successfully created.

The **ea** attribute format is set at file system creation. The **chfs** command can be used to convert the extended attribute format from **v1** to **v2**, but the format cannot be converted back. The conversion is done in an on-demand manner such that any extended attribute or ACL writes cause the conversion for that file object to occur.

Item	Description
-p *Prototype*	Specifies the name of the prototype file when you create a JFS file system. 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 must be used) or gigabytes (suffix G must be used). See JFS and JFS2 for more information.

Notes:

- The volume group in which the file system resides defines a maximum logical volume size and also limits the file system size.
- The **-s** *Size* flag specifies the minimum file size and cannot be decreased after the file system has been successfully created.
- The **maxext** attribute is ignored in older releases even if the filesystem was created with it on a later release .

Item	Description
-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.

Restriction: The **mkfs** command prevents EFS File System enablement of the following File Systems (mount points) because the security infrastructure (kernel extensions, libraries, and so on) are not available when you start the system. The following list is of known File Systems (mount points) that you cannot use:

"/"
"/usr"
"/var"
"/opt"

Security

Access Control: Only the root user or a member of the **system** group can run this command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 4 MB 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 i-node 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 4 MB file system on the **/dev/lv01** device with 512-byte fragments and one i-node for each 2048 bytes.
5. To create a JFS2 file system which can support NFS4 ACLs, type:

   ```
   mkfs -V jfs2 -o ea=v2 /dev/lv01
   ```

   This command creates an empty file system on the **/dev/lv01** device with **v2** format for extended attributes.

Files

Item	Description
/etc/vfs	Contains descriptions of virtual file system types.
/etc/filesystems	Lists the known file systems and defines their characteristics.

Related reference:

"mkproto Command" on page 768

Related information:

fsck command

filsys.h command

Understanding JFS Size Limitations

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 (whose length is administrator-configurable by way of the **chdev** command) 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 System Management Interface Tool (SMIT) **smit mkgroups** fast path to run this command.

The **mkgroup** command always checks the target group registry to make sure the ID for the new account is unique to the target registry. The **mkgroup** command can also be configured to check all group registries of the system using the **dist_uniqid** system attribute. The **dist_uniqid** system attribute is an attribute of the **usw** stanza of the **/etc/security/login.cfg** file, and can be managed using the **chsec** command.

The **dist_uniqid** system attribute has the following values:

- **never** - Does not check for ID collision against the non-target registries. This is the default setting.
- **always** - Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry account creation or modification fails.
- **uniqbyname** - Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the **dist_uniqid** system attribute.

The **uniqbyname** system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the **mkgroup** command is unspecified when creating a new account in a third registry using the colliding ID values. The new account creation might succeed or fail depending the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The **mkgroup** command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

If Encrypted File System (EFS) is enabled on the system, the **mkgroup** command updates the **/etc/security/group** file with EFS attributes (default values are added if you do not specify the attributes on the command line). If you do not specify `efs_keystore_access=none`, the **mkgroup** command creates the group keystore if at least one of the users has a keystore.

If the **mkgroup** command returns with the return code of 3, the keystore for the group is not created, but the **mkgroup** command creates the group.

Note: You can later create the group keystore by using the **efskeymgr** 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 multibyte 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:

Item	Description
:	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

Item	Description
-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.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.
To get the full functionality of the command, besides the **accessauths**, the role must also have the **aix.security.group.change** authorization. If Encrypted File System (EFS) is enabled on the system, the role must also have the **aix.security.efs** authorization to create the group keystore.

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**
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

Item	Description
/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 reference:

"mkuser Command" on page 839

Related information:

chgrpmem command

passwd command

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

Item	Description
-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

Item	Description
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:

gettable command

named command

gethostbyname command

hosts file

TCP/IP name resolution

mkiba Command

Purpose

Configures an IPv4 address or IPv6 address over the InfiniBand interface.

Syntax

mkiba { **-i** *Interface* **-a** *address* | **-v** *address6* **-A** *ib_adapter* **-p** *ib_port* [**-P** *P_KEY*] [**-V** *ipv6prefix*] [**-m** *subnet_mask*] [**-S** *state*] [**-M** *mtu*] [**-q** *queue_pair_size*] [**-Q** *Q_KEY*] [**-k** *superpacket*] }

Description

The **mkiba** command sets the minimal values required for using the IPv4 address or IPv6 address over the InfiniBand interface on a host machine. These values are written to the configuration database. This command also parses the information and verifies if the parameters are correct. If the interface is not defined, this command defines it, and calls the **chdev** command to configure it.

The following list details the functions of the **mkiba** command:

- Defining the interface name in the configuration database if it is not already defined
- Setting the host name in both the configuration database and the running machine
- Setting the IP address of the interface in the configuration database
- Setting the subnetwork mask, if applicable
- Setting the prefix length, if applicable
- Setting InfiniBand-specific parameters (such as host channel adapter (HCA), port, and so on)

You can use the **smit mkinetib** fast path in the System Management Interface Tool (SMIT) to run this command.

At least one IPv4 address or IPv6 address must be included to configure the interface.

Use the **ifconfig** ibX to check the state of the interface after the configuration.

The **ifconfig** command indicates that if the InfiniBand multicast group is pending, then there is an adapter malfunction or the physical port is down. If there is an adapter malfunction, you must perform manual HCA device driver reconfiguration to recover the InfiniBand stack. If the port is down, you must check the cabling and switches. If multicast is pending, check the InfiniBand Subnet Manager for errors. To troubleshoot the InfiniBand interface issues, use the **ibstat** command.

Flags

Item	Description
-A *ib_adapter*	Specifies the HCA. For example, iba0.
-a *address*	Sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host must have a unique Internet address. For example, a standard format for setting the Internet address is 127.10.31.2.
-i *Interface*	Specifies a particular InfiniBand interface. For example, ib0.
-k *superpacket*	Superpacket is a proprietary algorithm that allows the interface to receive large MTU (Maximum Transmit Unit) packets and fragment them in the interface layer to fit in the maximum MTU of the adapter. Enabling the superpacket automatically changes the *tcp_sendspace* attribute which specifies the number of bytes of data that the sending application can buffer in the kernel before the application is blocked on a send call, and the *tcp_recvspace* attribute which specifies the number of bytes of data that the receiving system can buffer in the kernel on the receiving sockets queue, with optimized values for the superpacket use.
-M *mtu*	Specifies the MTU for the interface. The IPv4 address or IPv6 address uses this MTU to fragment the packets.
-m *subnet_mask*	Specifies the mask that the gateway must use to determine the appropriate subnetwork for routing. The subnet mask is a set of 4 bytes. 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.
-P *P_KEY*	Specifies the partition key. The common partition keys are 0xFFFF and 0x7FFF.
-p *ib_port*	Specifies the HCA ports that you must use to configure the InfiniBand interface.
-q *queue_pair_size*	Specifies the size of the software queue. The range is from 256 to 32000.
-Q *Q_KEY*	Specifies the multicast qkey used to create a broadcast multicast group if there is no group previously created in the Subnet Manager. The common keys are 1, 0, and 0x1E.
-S *state*	Specifies whether the interface is active. When an interface is marked as inactive, any attempt to transmit messages through that interface fails. This action does not automatically disable routes using the interface.
-V *ipv6prefix*	Specifies the number of high-order bits used by routing protocols. The prefix is usually denoted following the IPv6 address and a slash (/). For example, the notation ff12::/16 represents a 16-bit prefix with a value of 1111111100010010.
-v *address6*	Specifies the IPv6 address. The address is a 128-bit address represented as eight 16-bit integers separated by colons. Each integer is represented by 4 hex digits. Leading zeros can be skipped, and consecutive null 16-bit integers can be replaced by two colons (one per address). For example, fe80:abcd:0000:0000:0000:0000:0260:8c2e:00a4.

Example

1. To set the required values to configure the IPv4 address over the InfiniBand interface, enter:
   ```
   mkiba -a 192.9.200.9 -i ib0 -A iba0 -p 1 -P -1 -q 4000 -M 2044 -m 255.255.255.0
   ```
2. To set the required values to configure the IPv6 address over the InfiniBand interface, enter:
   ```
   mkiba -v fe80::2:c903:1:1b40 -i ib0 -A iba0 -p 1 -P -1 -q 4000 -M 2044
   ```

Related information:

System Management: Communications and Networks

System Management Interface Tool: Overview

---

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 command-line prompting, template files can be created and edited directly by the user and passed to the **mkinstallp** command with the **-T** flag.

Flags

Item	Description
-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.

Note: Do not use the **.info** directory located in the *BaseDirectory* to store a template file. The template file may be removed when you run the **mkinstallp** command.

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

Item	Description
/usr/sbin/mkinstallp	Contains the **mkinstallp** command.

Related reference:

"installp Command" on page 62

mkiscsi Command

Purpose

Adds iSCSI target data.

Syntax

mkiscsi -l *AdapterName* **-g static -t** *TargetName* **-n** *PortNumber* **-i** *IPaddress* [**-p** *password*]

mkiscsi -l *AdapterName* **-g auto -t** *TargetName* **-p** *password*

mkiscsi -l *AdapterName* **-g** *group* **-f** *FileName*

Description

The **mkiscsi** command adds iSCSI target data to ODM. There are two categories of data stored in ODM. The first is for statically configured iSCSI targets, which require that all the relevant iSCSI target information (such as target name, IP address, and port number) are specified in order for AIX to discover them. The second category of iSCSI target data is for iSCSI target devices that can be configured automatically, but require authentication from the host (such as passwords). These two categories of iSCSI target data are associated with the **static** and **auto** groups, respectively, specified by the **-g** flag.

Flags

Item	Description
-f *FileName*	Specifies the filename from which iSCSI target information will be read and then placed into ODM.
-g *group*	Specifies which group this iSCSI target is associated with. There two valid groups are **static** and **auto**. The **static** group is for iSCSI targets that cannot be automatically discovered from this host; all relevant iSCSI target information for them (such as target name, IP address, and port number) must be specified. The **auto** group is for iSCSI targets that are automatically discovered, but require authentication information such as passwords.
-i *IPaddress*	Specifies the IP address of the iSCSI target.
-l *AdapterName*	Specifies the adapter name for the iSCSI TCP/IP Offload Engine (TOE) adapter that is attached to this iSCSI target. It can also specify the iSCSI protocol device for the iSCSI software solution device.
-n *NewPortNumber*	Specifies the port number on which the iSCSI target is accessed. The default port number is `3260`.
-p *password*	Specifies the new password for this iSCSI target.
-t *TargetName*	Specifies the iSCSI target name (for example, `iqn.sn9216.iscsi-hw1`).

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

The **mkiscsi** command is executable only by root.

Examples

1. To add one statically configured iSCSI target, enter the following command:
   ```
   mkiscsi -l ics0 -g static -t iqn.sn1234.iscsi_hw1
   ```
2. To add all the entries from the file **/etc/iscsi/targetshw**, enter the following command:
   ```
   mkiscsi -l ics0 -g static -f /etc/iscsi/targetshw
   ```

Location

/usr/sbin/mkiscsi

Files

Item	Description
src/bos/usr/sbin/iscsia	Contains the common source files from which the iSCSI commands are built.

Related information:

chiscsi command

rmiscsi command

---

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:

Item	Description
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

Item	Description
-i *Identifier*	Specifies which record in the **/etc/inittab** file the new record follows.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/etc/inittab	Contains the **mkitab** command.

Related information:

chitab command

init command

/etc/inittab file

---

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 System Management Interface Tool (SMIT) **smit mkkeyserv** fast path to run this command.

Flags

Item	Description
-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

Item	Description
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Related information:

smit command

startsrc command

Network File System (NFS) Overview for System Management

How to Start and Stop the Daemons

How to Export a File System Using Secure NFS

---

mkkrb5clnt Command

Purpose

Configures a Kerberos client.

Syntax

To configure Kerberos against IBM Network Authentication Service only:

mkkrb5clnt -h | **[-c** *KDC* **-r** *Realm* **-s** *Server* **-U [-a** *Admin* **] -d** *Domain* **[-A] [-i** *Database* **] [-K] [-T] [-t** *ticket_lifetime* **] [-n** *renew_lifetime* **]] [-l** {*ldapserver* | *ldapserver:port*} **]**

To configure Kerberos against non-kadmind services:

mkkrb5clnt -h | **-c** *KDC* **-r** *Realm* **-s** *Server* **-d** *Domain* **[-i** *Database* **] [-K] [-t** *ticket_lifetime* **] [-n** *renew_lifetime* **] –D [-l** {*ldapserver* | *ldapserver:port*} **]** | **-U**

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.

Item	Description
/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.

Item	Description
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

Item	Description
-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.
-D	Specifies Kerberos against non-kadmind services.
-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.
-n *renew_lifetime*	Specifies the client-specific time to generate a renewable ticket if the server supports it. By default, the ticket is nonrenewable. The *renew_lifetime* parameter value is composed of four numeric values that are delimited by colons.
-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 *ticket_lifetime*	Specifies the client-specific ticket lifetime for received tickets if the server supports it. If you do not specify the flag, the server sets the ticket lifetime. The *ticket_lifetime* parameter value is composed of four numeric values that are delimited by colons.
-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.

Item	Description
0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

A user with the **aix.security.kerberos** authorization is authorized to use this command.

Examples

1. To display the command syntax, enter the following command:

   ```
   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**, enter the following command:

   ```
   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, enter the following command:

   ```
   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
   ```

4. To configure **testbox.austin.ibm.com** as the client against a non-AIX machine, enter the following command:

   ```
   mkkrb5clnt -c non-aix.austin.ibm.com -r NON-AIX.AUSTIN.IBM.COM \
   -s non-aix.austin.ibm.com -d austin.ibm.com –D
   ```

5. To configure **testbox.austin.ibm.com** as the client against a non-AIX machine with the ticket lifetime of 1 day, 2 hours, 3 minutes, and 4 seconds, and the renew lifetime of 5 days, 6 hours, 7 minutes, and 8 seconds, enter the following command:

   ```
   mkkrb5clnt -c non-aix.austin.ibm.com -r NON-AIX.AUSTIN.IBM.COM \
   -s non-aix.austin.ibm.com -d austin.ibm.com –D \
   -t 1:2:3:4 -n 5:6:7:8
   ```

Files

Item	Description
/usr/krb5/sbin	Contains the **mkkrb5clnt** command.

---

mkkrb5srv Command

Purpose

Configures a Kerberos server.

Syntax

mkkrb5srv -h | [**-r** *Realm* **-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:

Item	Description
/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**.

Item	Description
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

Item	Description
-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.
-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.

Item	Description
0	Indicates the successful completion of the command.
1	Indicates that an error occurred.

Security

A user with the **aix.security.kerberos** authorization 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 -d austin.ibm.com
   ```

Files

Item	Description
/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

Item	Description
/usr/sbin/mklost+found	Contains the **mklost+found** command.

Related reference:
"mkfs Command" on page 716
Related information:
fsck command
Directories command
Files command

mklpcmd Command

Purpose

Defines a new least-privilege (LP) resource to the resource monitoring and control (RMC) subsystem and specifies user permissions.

Syntax

mklpcmd [**-n** *host*] [**-l**] [**-c 0** | **1** | **2** | **3**] [**–R** *RunCmdName*] [**-s** *FilterScript*] [**-A** *FilterArg*] [**-h**] [**-TV**] *resource_name command_path* [*ID perm*] ...

Description

The **mklpcmd** command defines a new LP resource to the resource monitoring and control (RMC) subsystem subsystem. An LP resource is a **root** command or script to which users are granted access based on permissions in the LP access control lists (ACLs). Specify the LP resource using the *resource_name* parameter. The *command_path* parameter specifies the command or script that could be run with LP access. Specify the complete path name of the command or the script. If *command_path* exists when a resource is created, the LP resource manager calculates the **CheckSum** and assigns the **CheckSum** attribute value. If *command_path* does not exist, the LP resource manager assigns **0** as the **CheckSum** attribute value.

Use the **-l** flag to lock the LP resource. The resource must be unlocked before it can be deleted. Use the **-c** flag to specify the control settings of the resource.

You can also use the **mklpcmd** command to specify permissions for users when you are creating a resource. To do this, you need to have administrator permission on the resources. Administrator permission gives you the ability to set and edit permissions. You can specify multiple user IDs and permissions with this command. See the **Examples** section for more information.

This command runs on any node. In a management domain or a peer domain, use the **-n** flag to define the LP resource on the node that is specified by *host*. Otherwise, this command runs on the local node.

Flags

-n *host* Specifies the node in the domain on which the LP resource is to be defined. By default, the LP resource is defined on the local node. The **–n** flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:
1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **mklpcmd** command runs once for the first valid scope that the LP resource manager finds.

-l Defines the new LP resource as locked so that it cannot be changed accidentally. The resource cannot be removed from the RMC subsystem until the **Lock** attribute is unset.

If you do not specify this flag, the new resource is not locked. This is the default.

-c 0 | 1 | 2 | 3
Sets the **ControlFlags** attribute, which is used to specify the control features for an LP command. If **ControlFlags** is not specified, it is set to **1** by default. Use this flag to specify one of these values:

0 Does not validate the **CheckSum** value.

1 Does not validate the **CheckSum** value. This is the default.

2 Validates the **CheckSum** value.

3 Validates the **CheckSum** value.

When an attempt is made to run the LP resource using the **runlpcmd** command, the value of the **ControlFlags** attribute determines which checks are performed before running the command represented by the resource.

In this release of RSCT, the **ControlFlags** attribute value specifies whether the **CheckSum** value is to be validated.

In previous releases of RSCT, the **ControlFlags** attribute value also specified whether the presence of certain characters in the input arguments to **runlpcmd** were to be disallowed. Checking for these characters is no longer necessary.

To maintain compatibility with LP resources that were defined in previous releases of RSCT, the **ControlFlags** attribute values, with respect to validating the **CheckSum** value, have remained the same. Consequently, values **0** and **1** indicate that the **CheckSum** value is not to be validated, and values **2** and **3** indicate that the **CheckSum** value is to be validated.

-R *RunCmdName*
Specifies the **RunCmdName** value for this resource, which will be used as a parameter of the **runlpcmd** command.

-s *script_path*
Specifies the fully-qualified path of the filter script.

-A *argument*
Specifies a string of arguments to be passed to the filter script.

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

-T Writes the command's trace messages to standard error.

-V Writes the command's verbose messages to standard output.

Parameters

resource_name
Is the name or identifier of the LP resource that is to be defined to the RMC subsystem.

command_path
Is the complete, fully-qualified path name of the command or script.

ID perm **...**
Specifies permissions for users when you are creating a resource. This parameter is optional.

ID Specifies the user identity for the ACL entry. See the **User identities** section of the **lpacl** information for the valid forms of this parameter.

perm Specifies the user permissions for the ACL entry. This parameter can consist of a combination of any of the following values:

r Read permission (consists of the **q**, **l**, **e**, and **v** permissions)

w Write permission (consists of the **d**, **c**, **s**, and **o** permissions)

a Administrator permission

x Execute permission

q Query permission

l Enumerate permission

e Event permission

v Validate permission

d Define and undefine permission

c Refresh permission

s Set permission

o Online, offline, and reset permission

0 No permission

See the **User permissions** section of the **lpacl** information for descriptions of these permissions.

Security

- To run the **mklpcmd** command with one or more *ID*:*perm* parameters, you need:
  - read and write permission in the Class ACL of the **IBM.LPCommands** resource class.
  - read and administrator permission in the Resource Initial ACL.

    As an alternative, the Resource Initial ACL can direct the use of the Resource Shared ACL if these permissions exist in the Resource Shared ACL.
- To run the **mklpcmd** command with no *ID*:*perm* parameters, you need write permission in the Class ACL of the **IBM.LPCommands** resource class.

Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT

Determines the system that is used for the session with the RMC daemon. 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 LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resource. The management scope determines the set of possible target nodes where the resource can be processed. The valid values are:

0 Specifies *local* scope.
1 Specifies *local* scope.
2 Specifies *peer domain* scope.
3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-V** flag is specified, this command's verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To create an LP resource called **LP1** that points to a command called **/tmp/user1/lpcmd1** on the local node, enter:
   ```
   mklpcmd LP1 /tmp/user1/lpcmd1
   ```
2. To create an LP resource called **LP2** that points to a command called **/tmp/my_command1** on **nodeB** in the management domain, enter:
   ```
   mklpcmd -n nodeB LP2 /tmp/my_command1
   ```
3. To create an LP resource called **lp3** with **ControlFlags** set to **3** (which means verify the **CheckSum** value), enter:
   ```
   mklpcmd -c 3 LP3 /tmp/cmd_lp3
   ```
4. To create an LP resource called **lp4** that points to **/tmp/testscript**, has a **RunCmdName** value of **test**, a **FilterScript** value of **/tmp/filterscr**, and filter arguments **node1** and **node2**, enter:
   ```
   mklpcmd -R test -f /tmp/filterscr -A "node1,node2" lp4 /tmp/testscript
   ```
5. To create an LP resource called **lp5** that points to **/usr/bin/mkrsrc** and gives users **user1@LOCALHOST** and **user2@LOCALHOST** read, write, and execute permission, enter:
   ```
   mklpcmd lp5 /usr/bin/mkrsrc user1@LOCALHOST rwx  user2@LOCALHOST rwx
   ```

Location

/opt/rsct/bin/mklpcmd
Contains the **mklpcmd** command

---

mklv Command

Purpose

Creates a logical volume.

Syntax

mklv [**-a** *position*] [**-b** *badblocks*] [**-c** *copies*] [**-C** *stripewidth*] [**-d** *schedule*] [**-R** *PreferredRead*] [**-e** *range*] [**-i**] [**-L** *label*] [**-m** *mapfile*] [**-o** *y / n*] [**-r** *relocate*] [**-s** *strict*] [**-t** *type*] [**-T O**] [**-u** *upperbound*] [**-v** *verify*] [**-w** *mirrorwriteconsistency*] [**-x** *maximum*] [**-y** *newlogicalvolume* | **-Y** *prefix*] [**-S** *stripsize*] [**-U** *userid*] [**-G** *groupid*] [**-P** *modes*] [**-p** *copyn=mirrorpool*] [**-O** *y* | *n*] *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.

The **-U**, **-G**, and **-P** flags can be used to set the ownership, group, and permissions, respectively, of the logical volume device special files. Only root users can set these values. For scalable and big vg format volume groups that are exported, specify the **-R** flag with the **importvg** command to restore these values upon import.

You can specify logical volumes sizes in 512 Blocks/KB/MB/GB when using the **mklv** command. The logical volumes sizes must be integer values.

Physical partitions are numbered starting at the outermost edge with number one.

Note:

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 **-C** or **-u** flag.
5. When creating a striped logical volume, the number of partitions must be an even multiple of the striping width. If not, the number of partitions will be rounded up to the next valid value.
6. The **mklv** command is not allowed on a snapshot volume group.
7. Mirror Write Consistency (MWC) and Bad Block Relocation (BBR) are not supported in a concurrent setup with multiple active nodes accessing a disk at the same time. These two options must be disabled in this type of concurrent setup.
8. Bad block relocation policy of a logical volume is not supported on a volume group that is created with 4 KB block physical volumes.

You can 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 file system on a striped logical volume, you should create the striped logical volume before you run the **crfs** command or **mkfs** command to create the file system. 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

Item	Description
-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.
-C *stripewidth*	Sets the Stripe width of the logical volume. If the *Stripewidth* is not entered it is assumed to be the *upperbound* or the total number of disks specified on the command line.
-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 robin 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. **Note:** The **-R** flag overwrites the read policy specified by the **-d** flag. If the preferred copy is not available, the read operations follow the scheduling policy.
-R *PreferredRead*	Sets read preference to the copy of the logical volume. If the **-R** flag is specified and if the preferred copy is available, the read operation occurs from the preferred copy. If the preferred copy is not available, the read operations follow the scheduling policy of the logical volume. The *PreferredRead* variable can be set to a value in the range 0 -3. The default value is 0.

Item	Description
-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 by the file designated by the *mapfile* parameter. 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. **Important:** When you use map files, you must understand and adhere to all LV-allocation parameters such as strictness, upperbound, and stripe width. Using map files bypasses the checks done in the LVM-allocation routines. This is important for striped LVs, which are assumed to have a typical striped allocation pattern conforming to the stripe width.
-o *y* / *n*	Turns on/off serialization of overlapping I/Os. If serialization is turned on then overlapping I/Os are not allowed on a block range and only a single I/O in a block range is processed 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`.
-O *y* / *n*	Enables the infinite retry option of the logical volume. **n** The infinite retry option of the logical volume is not enabled. The failing I/O of the logical volume is not retried. This is the default value. **y** The infinite retry option of the logical volume is enabled. The failed I/O request is retried until it is successful. **Note:** The infinite retry option is ignored for a logical volume (LV) when *active* mirror write consistency is set. The infinite retry option must be enabled at the volume group level to work for a logical volume when *active* mirror write consistency is set. **Note:** The infinite retry option is not supported in the Geographic Logical Volume Manager (GLVM) environment.
-p *copyn*=*mirrorpool*	Enables mirror pools for the logical volume. A mirror pool is assigned to a copy using the *copyn*=*mirrorpool* parameter. Specify a mirror pool for each copy. To specify more than one *copyn*=*mirrorpool* pair, provide multiple **-p** *copyn*=*mirrorpool* flags. Mirror pool names can be up to 15 characters and follow the same rules that apply to volume group names and logical volume names.
-P *modes*	Specifies permissions (file modes) for the logical volume special file.

Item	Description
-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 *stripSize*	Specifies the number of bytes per strip (the strip size multiplied by the number of disks in an array equals the stripe size). Valid values include 4K, 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M, 32M, 64M, and 128M. **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 file system, 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.
-T O	The **`-T O`** option indicates that the logical volume control block does not occupy the first block of the logical volume. Therefore, the space is available for application data. Applications can identify this type of logical volume with the **`IOCINFO`** `ioctl` operation. The logical volume has a device subtype of `DS_LVZ`. A logical volume created without this option has a device subtype of `DS_LV`.
-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 super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *stripewidth*. If *upperbound* is not specified it is assumed to be *stripewidth* for striped logical volumes.
-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.

Item	Description
-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 function is available only on **big** type and **scalable** type of 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.
-y *newlogicalvolume*	Specifies the logical volume name rather than having the name generated automatically. Logical volume names must be unique system wide and can range from 1 to 15 characters. If the *volumegroup* is varied on in concurrent mode, the new logical volume name should be unique across all the concurrent nodes where the *volumegroup* is varied on. The name cannot begin with a prefix already defined in the **PdDv** class in the Device Configuration Database for other devices. The logical volume name created is sent to standard output. The logical volume name can only contain the following characters: • "A" through "Z" • "a" through "z" • "0" through "9" • "_" (the underscore) • "-" (the minus sign) • "." (the period) All other characters are considered not valid.
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 strip 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 strip 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
   ```

8. To create three copies of logical volume in volume group `testvg` and to set the read preference to the second logical volume copy, enter the following command:

   ```
   mklv -c 3 -R 2 -y testlv testvg 10
   ```

Files

Item	Description
/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:

chfs command

extendlv command

chlv command

Logical volume storage

mklvcopy Command

Purpose

Provides copies of data within the logical volume.

Syntax

mklvcopy [**-a** *position*] [**-e** *range*] [**-k**] [**-m** *mapfile*] [**-s** *strict*] [**-u** *upperbound*] [**-p** *copyn=mirrorpool*] *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:

- To use this command, you must either have `root` user authority or be a member of the **system** group.
- The **mklvcopy** command is not allowed on a snapshot volume group.
- When you create a copy of a logical volume with a superstrict allocation policy, the **mklvcopy** command first attempts to mimic the physical-partition mapping of the first mirror copy onto another set of disks in the volume group. This algorithm ignores the interphysical and intraphysical volume allocation policies, even when the policies are specified as arguments to the **mklvcopy** command. If it is not possible to mimic the first copy's physical partition mapping, the usual allocation algorithm, which utilizes the interphysical and intraphysical volume allocation policies, is used.

You can use the System Management Interface Tool (SMIT) **smit mklvcopy** fast path to run this command.

Flags

Note: The **-e** and **-s** flags are not valid with a striped logical volume.

Item	Description
-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.

Item	Description
-m *mapfile*	Specifies the exact physical partitions to allocate. Partitions are used in the order given by the file designated by the *mapfile* parameter. 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. **Important:** When you use map files, you must understand and adhere to all LV-allocation parameters such as strictness, upperbound, and stripe width. Using map files bypasses the checks done in the LVM-allocation routines. This is important for striped LVs, which are assumed to have a typical striped allocation pattern conforming to the stripe width.
-p *copyn=mirrorpool*	Assigns mirror pools to the copies being created. A mirror pool is assigned to a copy using the *copyn=mirrorpool* parameter. Specify a mirror pool for each copy being created. To specify more than one *copyn=mirrorpool* pair, provide multiple **-p** *copyn=mirrorpool* flags.
-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. See Note 4 for other effects of the superstrict allocation policy on **mklvcopy** behavior. **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 maximum number of physical volumes per VG. When using super strictness, the upper bound indicates the maximum number of physical volumes allowed for each mirror copy. When using striped logical volumes, the upper bound must be multiple of *stripewidth*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/sbin/mklvcopy	Contains the **mklvcopy** command.

Related reference:
"mklv Command" on page 739
Related information:
syncvg command
varyonvg command

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 System Management Interface Tool (SMIT) **smit mkmaster** fast path to run this command.

Flags

Item	Description
-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

Item	Description
/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:

chmaster command

ypinit command

ypbind command

yppasswdd command

---

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 System Management Interface Tool (SMIT) **smit mknamerslv** fast path to run this command.

Flags

Item	Description
-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`. The **"attribute=value"** pairs must not be placed in one line. As an example, enter the **"attribute=value"** pairs to `namsv.file` in the following format:

   ```
   domain=austin.century.com
   nameserver=192.9.200.1
   ```

Files

Item	Description
/etc/resolv.conf	Contains domain name server information for local resolver routines.

Related information:

namerslv command

TCP/IP name resolution

Transmission Control Protocol/Internet Protocol

resolv.conf file

---

mknetid Command

Purpose

Generates data for the **netid.byname** map for use by the Network Information Services (NIS).

Syntax

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

Item	Description
-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

Item	Description
/var/yp/Makefile	**mknetid** is most commonly used when rebuilding the NIS databases using **/var/yp/Makefile**.
/etc/passwd	Where *PasswordFile* resides.
/etc/groups	Where *GroupFile* resides.
/etc/hosts	Where *HostsFile* resides.
/etc/netid	Where *NetidFile* resides.

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

Item	Description
-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

Item	Description
inittab	Controls the initialization process of the system.
/etc/rc.nfs	Contains the startup script for the NFS and NIS daemons.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"mknfsexp Command" on page 752

“mknfsmnt Command” on page 755

Related information:

chnfs command

rmnfs command

mknfsexp Command

Purpose

Exports a directory to NFS clients.

Syntax

/usr/sbin/mknfsexp -d *Directory* [**-e** *pathname*] [**-f** *Exports_File*] [**-t** [{ **rw** | **ro** | **remove** } { **rm -h** *HostName* [**,** *HostName ...*] }]] [**-a** *UID*] [**-r** *HostName* [**,** *HostName ...*]] [**-c** *HostName* [**,** *HostName ...*]] [**-s** | **-n**] [**-I** | **-B** | **-N**] [**-P** | **-p**] [**-D** {yes | no}] [**-v**] [**-x** | **-X**] [**-S** *flavor* [*,flavor*]] [**-G** *rootpath@host*[**+***host*][**:***rootpath@host*[**+***host*]]] [**-g** *rootpath@host*[**+***host*][**:***rootpath@host*[**+***host*]]] [**-o** *Ordering*]

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

Item	Description
-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.
-B	Adds an entry to the **/etc/exports** file and the **exportfs** command is executed to export the directory. This flag is the default.
-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.
-d *Directory*	Specifies the directory that is to be exported or changed.
-D {yes | no}	Enables or disables file delegation for the specified export. This option overrides the system-wide delegation enablement for this export. The system-wide enablement is done through **nfso**.
-e *pathname*	Specifies an export name for the directory.
-f *Exports_File*	Specifies the full path name of the exports file to use if other than the **/etc/exports** file.
-g *rootpath@host* [*+host*] [*:rootpath@host* [*+host*]]	The specified directory will be marked with replica information. If the server becomes unreachable by an NFS client, the client can switch to one of the specified servers. This option is only accessible using NFS version 4 protocol, and version 4 access must be specified in the options. Because the directory is being exported for client access, specifying NFS version 2 or version 3 access will not cause an error, but the request will simply be ignored by the version 2 or version 3 server. This option cannot be specified with the **-G** flag. Only the host part of each specification is verified. The administrator must ensure that the specified *rootpaths* are valid and that the target servers contain appropriate data. If the directory being exported is not in the replica list, that directory will be added as the first replica location. The administrator should ensure that appropriate data exists at the replica locations. For a more complete description of replication, see the **exportfs** command. The **-g** option is available only on AIX 5.3 with 5300-03 or later. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use **chnfs -R on** to enable replication.

Item	Description
-G *rootpath@host* [*+host*] [*:rootpath@host* [*+host*]]	A namespace referral will be created at the specified path. The referral directs clients to the specified alternate locations where they can continue operations. A referral is a special object. If a nonreferral object exists at the specified path, the export is disallowed and an error message is printed. If nothing exists at the specified path, a referral object is created there that includes the path name directories leading to the object. A referral cannot be specified for the **nfsroot**. The name `localhost` cannot be used as a *hostname*. The **-G** option is allowed only for version 4 exports. If the export specification allows version 2 or version 3 access, an error message will be printed and the export will be disallowed. The administrator should ensure that appropriate data exists at the referral locations. For a more complete description of referrals, see the **exportfs** command. The **-G** option is available only on AIX 5L™ Version 5.3 with the 5300-03 Recommended Maintenance package or later. **Note:** A referral or replica export can only be made if replication is enabled on the server. Use **chnfs -R on** to enable replication.
-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.
-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 require the client to use the more secure protocol. This flag is the default.
-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.
-o *Ordering*	Defines how the alternate locations list is generated from the servers that you specified on the **refer** or **replicas** option. The option applies only to directories exported for access by NFS version 4 protocol. The *Ordering* parameter has three allowable values: **full** All of the servers are scattered to form the combinations of alternate locations. **partial** The first location of all the combinations is fixed to the first server specified on the **refer** or **replicas** option. The rest of the locations and the first location are scattered as if they are scattered using the `scatter=full` method. **none** No scatter is to be used. The value can also be used to disable scattering if it was enabled previously.
-p	Specifies that the exported directory is not a public directory.
-P	Specifies that the exported directory is to be a public directory.
-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.
-s	Requires clients to use a more secure protocol when accessing the directory.
-S *flavor*[*,flavor*]	May be used in conjunction with the **-c**, **-t**, or **-r** options to associate the option with one or more specific security methods. Most **exportfs** options can be clustered using the **sec** option. Any number of **sec** stanzas may be specified, but each security method can be specified only once. Allowable flavor values are: **sys** UNIX authentication. **dh** DES authentication. **none** Use the anonymous ID if it has a value other than -1. Otherwise, a weak auth error is returned. **krb5** Kerberos. Authentication only. **krb5i** Kerberos. Authentication and integrity. **krb5p** Authentication, integrity, and privacy.
-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. **remove** Removes the exported directory. **rm** Exports the read-mostly directory. If chosen, the **-h** flag must be used to specify the hosts that have read-write permission.
-v *number* [**,** *number*] ...	The directory specified by the **-d** option is made available to clients using the specified NFS versions. Valid values are 2, 3, or 4. You can export two entries for the same directory with different versions 2 (or 3) and 4.

Item	Description
-x	Accepts the replica location information specified with the **-g** option as-is. Does not insert the server's primary hostname into the list if it is not present. This flag is intended for use with servers with multiple network interfaces. If none of the server's host names are in the replica list, NFSv4 clients might treat the location information as faulty and discard it.
-X	Enables auto-insert of the primary hostname into the replica list. If the server's primary hostname is not specified in the replica list, the hostname will be added as the first replica location.

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.
5. To export the **/common/documents** directory to allow access only to clients using NFS version 4 protocol, enter:

   ```
   mknfsexp -d /common/documents -v 4
   ```
6. To export the **/common/documents** directory, allowing access to `client1` and `client2` for clients using krb5 access, enter:

   ```
   mknfsexp -d /common/documents -S krb5 -r client1,client2
   ```
7. To export the **/common/documents** directory with full scattering for the hosts named `s1` and `s2` specified as referrals, enter the following command:

   ```
   mknfsexp -d /common/documents -v 4 -G  /common/documents@s1:/common/
   documents@s2 -o full
   ```
8. To export the **/common/documents** directory with partial scattering at hosts named `s1`, `s2` and `s3`, specified as replicas, enter the following command:

   ```
   mknfsexp  -d /common/documents -v 4 -g /common/documents@s1:/common/
   documents@s2:/common/documents@s3 -o partial
   ```
9. To export the **/common/documents** directory with the export name `/exports1/cool/mike`, enter the following command:

   ```
   mknfsexp -d /common/documents -e /exports1/cool/mike -S sys -v 4
   ```

Files

Item	Description
/etc/exports	Lists the directories that the server can export.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related information:

chnfsexp command

exportfs command

rmnfsexp command

NFS Installation and Configuration

---

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**] [**-I** | **-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* | *4* }] [**-k** { *any* | *tcp* | *udp* }] [**-l** | **-L**] [**-M** *security_methods*] [**-i** { *dio* | *cio* [*,cior*] }]

Description

The **mknfsmnt** command constructs an entry that is appended to the **/etc/filesystems** file, thus making a file system available for mounting. If the mount is to be permanent, this entry remains. If the mount is temporary, the flags are used directly for the **mount** command. If the mount is soft and the server does not respond, the system returns an error. If the mount is hard, the client continues trying until the server responds. The hard mount is the default mount.

Flags

Item	Description
-A	The **/etc/filesystems** entry for this file system specifies that it should be automatically mounted at system restart.
-a	The **/etc/filesystems** entry for this file system specifies that it should not be automatically mounted at system restart. This is the default flag.
-B	Adds an entry to the **/etc/filesystems** file and attempts to mount the file system. This is the default flag.
-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 flag.
-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.

Item	Description
-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 flag.
-H	Creates a hard mount, which causes the client to continue retrying until the server responds. This is the default flag.
-h *RemoteHost*	Specifies the NFS server that is exporting the directory.
-I	Causes an entry to be added to the **/etc/filesystems** file. The directory is not mounted.
-i	Specifies I/O mode for the mount. The options are: *cio* Specifies concurrent I/O mode. Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. *dio* Specifies direct I/O mode. Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call. *cior* Specifies concurrent I/O with read-only mode. **Note:** For more information on the **cio** and **dio** options, see the **mount** command.
-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 flag.
-K	Specifies the NFS version used for this NFS mount. The options are: *any* Uses the **mount** command to determine the correct match. Refer to the **mount** command for a description of the current default behavior. **2** Specifies NFS Version 2. **3** Specifies NFS Version 3. **4** Specifies NFS Version 4.
-k	Specifies the transport protocol used for the mount. The 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.
L	Indicates that the lock requests are handled locally without connecting to the server.
l	Indicates that the lock requests are not handled locally. The server handles the lock requests.
-M *security_methods*	A list of security methods to use when attempting the mount. A comma separated list of the values `sys`, `dh`, `krb5`, `krb5i`, `krb5p`, which correspond to UNIX, DES, Kerberos 5, Kerberos 5 with integrity, and Kerberos 5 with privacy. Multiple values are allowed, but are meaningful only with NFS Version 4 mounts. If multiple methods are given for a Version 2 or 3 protocol mount, the first method is used. For a NFS Version 4 mount, the methods are tried in the listed order.
-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 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 is the default flag.
-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. The default number of daemons is `7` for NFS Version 2 and `32` for NFS Version 3 and NFS Version 4.
-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 flag.
-r *TimesToRetry*	Indicates the number of times to retry a mount. The default value is 1000.

Item	Description
-R *NumRetrans*	For a soft mount, this flag specifies the number of times that a request has to be transmitted if it is not acknowledged by the server. If the request is unacknowledged after *NumRetrans* transmissions, the client gives up the request. If this flag is not specified, the default value 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 the 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 flag.
-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 flag.
-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 flag.

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

Item	Description
/etc/filesystems	Lists the remote file systems to be mounted during the system restart.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"mount Command" on page 885

Related information:

chnfsmnt command

rmnfsmnt command

How to Mount a NFS File System Explicitly

mknfsproxy Command

Purpose

Creates a new NFS proxy-enabled Cachefs instance that is backed with an NFS client mount to a back-end NFS server.

Syntax

/usr/sbin/mknfsproxy [**-L**] **-c** *local_cache_directory* **-d** *Cachefs_mount_point* [**-o param=n**[**, param=n**]] **-m** [*nfs_mount_options*] *remote_server*:*remote_directory* [**-e** [*export_option*, [*export_option*]]]

Description

The local file system used by the created Cachefs instance must be a JFS2 file system. The required inputs include the remote server and directory (*remote_server*:*remote_directory*) that the Cachefs instance will access, the local directory (*local_cache_directory*) where information will be cached, and the directory where the Cachefs will be mounted.

After the cache is initialized, the Cachefs instance is mounted and ready to be NFS exported. Provide NFS export information so that the cached view will also be NFS exported using the specified options.

Flags

Item	Description
-c	Specifies the local JFS2 file system directory where Cachefs will store cached data and state. This is a required option.
-d	Specifies the directory where Cachefs will be mounted. This is a required option.
-e	Specifies the NFS server export options for the created Cachefs instance. If this is supplied, the created Cachefs instance will also be NFS exported using the supplied options. If this option is not supplied, the created Cachefs instance will be exported with the same NFS version specified by the **-m** option.
-L	Causes the Cachefs instance to acquire a single lock from its associated NFS back-end that covers the entire file when any byte range locks are requested. When the count of byte range locks drops to 0 (zero), the lock on the back-end NFS server is released.

Item	Description
-m	Specifies the NFS client mount, which might optionally include NFS client mount options as described in the **mount** man page. This is a required option, and the remote server and remote directory must be supplied.
-o	Specifies Cachefs configuration options in the form **param=n**. For descriptions of the Cachefs resource parameters, refer to the **cfsadmin** command.

Parameters

Item	Description
Cachefs_mount_point	Specifies where the proxy-enabled Cachefs instance is to be mounted.
export_option	Specifies which options of the **export** command are used for the Cachefs instance.
local_cache_directory	Specifies the local directory where information is cached.
nfs_mount_options	Specifies the NFS client options of the **mount** command.
remote_directory	Specifies the remote directory that the Cachefs instance accesses.
remote_server	Specifies the remote server that the Cachefs instance accesses.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Examples

1. To create a proxy-enabled Cachefs instance at `/edge` that accesses `/project1` at NFS server `foo`, enter:

   ```
   mknfsproxy -c /cache/project1 -d /edge -m hard,vers=4,intr foo:/project1
   ```

   In the preceding example, information accessed through `/edge` is cached at `/cache/project1`.

   The following variation of the preceding example also exports the created Cachefs instance (`proj1_cached`) for NFS V4 access with authentication flavor of Kerberos 5 and an external name set to `/nfs4/projects/project1`:

   ```
   mknfsproxy -c /cache/project1 -d /edge -m hard,vers=4,intr foo:/project1
    -e sec=krb5,vers=4,exname=/nfs4/projects/project1
   ```

Location

/usr/sbin/mknfsproxy

Related information:

cfsadmin command

rmnfsproxy command

---

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 the major and minor numbers be defined in this object class to ensure 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

Item	Description
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).

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/sbin/mknod	Contains the **mknod** command.

Related information:

mknod command

List of Device Configuration Subroutines

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

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC **Subsystem Configuration** object class.
/etc/objrepos/SRCnotify	Specifies the SRC **Notify Method** object class.

Related reference:

"mkssys Command" on page 825

Related information:

rmnotify command

System Resource Controller

Defining Your Subsystem to the SRC

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:

Item	Description
/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

Item	Description
-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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Item	Description
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

Item	Description
/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 reference:
"mkuser Command" on page 839
Related information:
passwd command
pwdadm command
getpwnam command

mkpath Command

Purpose

Adds to the system another path to an MPIO capable device.

Syntax

mkpath [**-l** *Name*] [**-p** *Parent*] [**-w** *Connection*] [**-i** *PathID*]

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

Item	Description
-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.
-i *PathID*	Indicates the path ID associated with the path to be added and is used to uniquely identify a path. This flag cannot be used with the **-d** flag.
-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

Item	Description
/usr/sbin/mkpath	Contains the **mkpath** command.

Related reference:
"lspath Command" on page 474

mkprojldap Command

Purpose

Configures the LDAP client and server machines for handling advanced accounting subsystem data.

Syntax

mkprojldap -s -h *hostname* **-D** *bindDN* **-w** *bindPWD* **-i -p** *projectInstallPoint* **-a** *adminInstallPoint*

mkprojldap -u -h *hostname* **-D** *bindDN* **-w** *bindPWD*

mkprojldap -c -D *bindDN* **-w** *bindPWD* [**-p** *accountingProjectDN*] [**-a** *accountingAdminDN*] [**-r** *cron*]

mkprojldap { **-l** | **-L** [**-D** *bindDN* **-w** *bindPWD*] | **-V** } [**-p**] [**-a**]

Description

The **mkprojldap** command configures the LDAP server and client machines for handling the advanced accounting subsystem data. The LDAP server and client relationship must already be defined, and **mkprojldap** makes only incremental changes. The **mkprojldap** command can be used to configure the basic LDAP connection.

To add advanced accounting support to the LDAP server, the LDAP schema for advanced accounting must be uploaded to the server. The schema describes the format of advanced accounting data to the server, enabling the server to process accounting data without being enabled specifically for accounting. This is accomplished with the **-u** option. The LDAP server is not dependent on advanced accounting. This command needs to be run only once for each LDAP server. After this command is run, use the **-s** option to define the location on the LDAP server where advanced accounting data is to be stored. This command can be run one or more times to establish one or more accounting domains. An LDAP client can only access only one accounting domain at a time.

To configure an LDAP client so that it receives advanced accounting data, use the **-c** option to specify the location of the advanced accounting data sets on the LDAP server that are to be used by the LDAP client. The **mkprojldap** command is used to configure absolute paths, which are known as *distinguished names* (DNs), to projects and admin policies. The advanced accounting subsystem stores project definitions and admin policies on LDAP servers, so there are two advanced accounting DNs that can be configured. The **mkprojldap -c** command must be run on each client.

Flags

Item	Description
-a *accountingAdminDN*	Specifies the accounting admin DN location on the LDAP server, when used with **-s** or **-c** options. When used with **-l** or **-L** options, this flag displays the accounting admin DN.
-c	Configures the LDAP client.
-D *bindDN*	Specifies the Bind DN to be used during the server configuration.
-h *hostname*	Specifies the host name of the LDAP server during the server configuration.
-i	Provides the admin (**-a**) and project (**-p**) install points during the server configuration.
-L	Displays the potential accounting DNs that are visible from the server.
-l	Displays the accounting DNs in the **ldap.cfg** file.
-p *accountingProjectDN*	Specifies the accounting project DN location on the LDAP server when used with the **-s** or **-c** options. When used with **-l** or **-L** options, this flag displays the accounting project DN.
-r *con*	Specifies the frequency for refreshing the LDAP repositories (hourly, daily, or off).
-s	Configures the LDAP server.
-u	Uploads the advanced accounting schema to the LDAP server.
-V	Displays the current LDAP client configuration details in a colon separated format.
-w *bindPWD*	Used to provide the Bind password for the Bind DN specified with the **-D** option.

Note: When using the preceding flags with this command, use the following guidelines:

- During server and client configuration, both the **-p** and **-a** arguments can be specified at the same time, but neither is required. If neither is specified, the **mkprojldap** command tries to compute the missing accounting DNs by searching for the objects on the LDAP server. These objects are **ou=projects** and **ou=adminpolicy**. If an object is found, the corresponding accounting DN is computed and added to the **ldap.cfg** file.
- While listing the accounting DNs using the **-l** or **-L** options, both **-p** and **-a** can be used. If neither of them are provided, all accounting DNs in the **ldap.cfg** file are listed.
- The colon-separated data displayed by the **-V** option takes the following format:

  ```
  ldap-server-hostname:bind DN:bind password:default-projectdn:default-admindn:cron
  ```

Exit Status

Item	Description
0	Successful completion.
>0	An error occurred.

Examples

1. To upload the advanced accounting schema, type:

   ```
   mkprojldap -u -h mozilla -D cn=root -w mozillapasswd
   ```

2. To configure the LDAP server, type:

   ```
   mkprojldap -s -h ldap.svr.com -D cn=root -w passwd -i
               -p cn=aixdata,o=ibm -a cn=aixdata,o=ibm
   ```

   This command creates two DNs in the following format:

   ```
   ou=projects,ou=aacct,cn=aixdata,o=ibm and ou=adminpolicy,ou=aacct,cn=aixdata,o=ibm
   ```

3. To configure the LDAP client, type:

   ```
   mkprojldap -c -D cn=testroot -w testpwd -p ou=projects,ou=aacct,ou=cluster1,cn=aixdata -a
              ou=adminpolicy,ou=aacct,ou=cluster1,cn=aixdata -r hourly
   ```

4. To display the currently configured accounting DNs, type:
   ```
   mkprojldap -l
   ```

Files

Item	Description
/usr/sbin/mkprojldap	Contains the **mkprojldap** command.
/etc/security/ldap/ldap.cfg	Contains the LDAP configuration data.
/etc/security/ldap/sec.ldif	Contains the LDAP schema for advanced accounting.

Related information:

projctl command

secldapclntd 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:

Prototype Files

Item	Description
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:

Tokens

Item	Description
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.

Tokens

Item	Description
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:

dir2 object

Item	Description
`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

Files

Item	Description
/usr/sbin/mkproto	Contains the **mkproto** command.

Related reference:

"mkfs Command" on page 716

Related information:

fsdb command

proto command

filsys.h file

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 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

Item	Description
-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

Item	Description
-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

Item	Description
?	Displays usage information for the **mkprtldap** command.

Security

This command can be run by the `root` user only.

Examples

1. 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
   ```

2. 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**).

3. 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.

4. 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.

5. 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 Version 5.2) configuration information.
rw	**/home/ldapdb2/idsslapd-ldapdb2/etc/ibmslapd.conf**	(Server configuration) - Contains the IBM Directory (LDAP Version 6.0 or later) 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 reference:

"lp Command" on page 266

"lpstat Command" on page 331

"mksecldap Command" on page 809

Related information:

Trusted AIX®

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

Item	Description
-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.

Item	Description
-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

Item	Description
/etc/lpd/qconfig	Contains configuration information for the printer queuing system.

Related reference:

"mkque Command" on page 780

"mkquedev Command" on page 782

Related information:

startsrc command

TCP/IP daemons

---

mkps Command

Purpose

Adds an additional paging space.

Syntax

To Add a Logical Volume for Additional Paging Space

mkps [**-t** *lv | [ps_helper psname]*] [**-a**] [**-n**] [**-c** *ChksumSize*]**-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.

Note: A paging space larger than 2 GB is possible when using NFS (Network File System) v4 rather than the default UDP (User Datagram Protocol) or the NFSv2 protocol. NFSv2 swapping can only handle swap file size up to 2 GB due to NFSv2 protocol limitation.

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.

When adding a NFS paging space, the client attempts to contact the server using UDP and then TCP. The method that succeeds first is used to contact the server when accessing that paging space.

If the **-t** flag is specified, the argument will be assumed to be a third-party helper executable. If the helper executable is present in the /sbin/helpers/pagespace path then it will be spawned passing all the arguments and with the **-m** flag to specify **mkps** command. An entry will be added into /etc/swapspaces path if the helper executable returns zero. In this case, if *psname* starts with /, it is considered to be absolute path of device entry , or else /dev is prepended to the *psname*. The helper executable must take care of creating the device, making it pageable and adding an entry into ODM. If the helper program doesn't exist in the /sbin/helpers/pagespace directory the **mkps** command will display the usage error. The helper executable must exit with a 0 if successful and a non-zero if it fails.

You can use the System Management Interface Tool (SMIT) **smit mkps** fast path to run this command.

Flags

Item	Description
-a	Specifies that the paging space is configured at subsequent restarts.
-c	Specifies the size of the checksum to use for the paging space, in bits. Valid options are 0 (checksum disabled), 8, 16 and 32. If -c is not specified it will default to 0.
-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. *ps_helper* Name of the helper program for a third party device. *psname* Name of the device entry for paging space.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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
   ```

3. To create a paging space myps using helper executable foo:

   ```
   mkps -t foo /dev/myps -s1 myvg mydisk
   ```

Files

Item	Description
/etc/swapspaces	Specifies the paging space devices and their attributes.

Related reference:
"mklv Command" on page 739
Related information:
rmps command
swapon command
Logical volume storage

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

Item	Description
-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

Item	Description
inittab	Controls the initialization process of the system.
/etc/rc.qos	Contains the startup script for the QoS daemons.

Related information:
rmqos command
QoS installation
TCP/IP Quality of Service

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 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
```

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.
If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/mkque	Contains the **mkque** command.
/etc/qconfig	Configuration file.

Related reference:
"mkquedev Command"
Related information:
qconfig file
Printing administration

---

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 System Management Interface Tool (SMIT) **smit mkquedev** fast path to run this command.

Recommendation: To edit the **/etc/qconfig** file, use the **chque**, **mkque**, **rmque**, **chquedev**, **mkquedev**, and **rmquedev** commands or SMIT. Further, it is recommended to run these commands during slow or off-peak time.
If manual editing of the **/etc/qconfig** file is necessary, you can first issue the **enq -G** command to bring the queuing system and the **qdaemon** to a halt after all jobs are processed. Then you can edit the **/etc/qconfig** file and restart the **qdaemon** with the new configuration.

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/mkquedev	Contains the **mkquedev** command.
/etc/qconfig	Contains the configuration file.

Related reference:

"lsquedev Command" on page 488

Related information:

chquedev command

/etc/qconfig file

Installing support for additional printers

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. A suffix of *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.

By default, RAM disk pages are pinned. Use the **-u** flag to create RAM disk pages that are not pinned.

Flags

Item	Description
-u	Specifies that the ram disk that is created will not be pinned. By default, the ram disk will be pinned.

Parameters

Item	Description

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:

Item	Description
0	Successful completion.
>0	An error occurred.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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
```

4. To set up a RAM disk that is approximately 20 MB in size and create a JFS file system on that RAM disk, enter the following commands:

```
mkramdisk 40000
ls -l /dev | grep ram
mkfs -V jfs /dev/ramdiskx
mkdir /ramdisk0
mount -V jfs -o nointegrity /dev/ramdiskx /ramdiskx
```

x is the logical RAM disk number.

To set up a RAM disk that is approximately 20 MB in size and create a JFS2 file system on that RAM disk, enter the following commands:

```
mkramdisk 40000
ls -l /dev | grep ram
/sbin/helpers/jfs2/mkfs -V jfs2 /dev/ramdiskx
mkdir /ramdiskx
mount -V jfs2 -o log=NULL /dev/ramdiskx /ramdiskx
```

x is the logical RAM disk number.

Note: For both JFS and JFS2, for using a file system on a RAM disk, the RAM disk must be pinned.

Files

Item	Description
/usr/sbin/mkramdisk	Contains the **mkramdisk** command.

Related information:
rmramdisk command

mkresponse Command

Purpose

Creates a new response definition.

Syntax

To create a response with no actions:

mkresponse [**-b**] [**-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*] [**-b** | [**-e a** | **A** | **b** | **e** | **r**]] [**-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.

Use the **-b** flag to specify that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched. The **-b** flag cannot be specified with the **-e** flag.

In a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the response definition. If you are using **mkresponse** on the management server and you want the response to be defined on the management server, do *not* specify the **-p** flag. If the **-p** flag is not specified, the response is defined on the local node.

Use the **chresponse** command to add actions to a response or to remove actions from a response. Use the **startcondresp** command to start monitoring. The **startcondresp** command links a response to a condition, if they are not already linked.

To lock a response so it cannot be modified or removed, use the **chresponse** command with the **-L** flag.

Flags

-b Specifies that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched.

An event response can be created for batched event conditions without an action script.

The **-b** flag cannot be specified with the **-e** flag.

-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 cluster. 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 response as indicated by the flags.

-d *days_of_week*

Specifies the days of the week when the 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 values for each day follow:

1 Sunday
2 Monday
3 Tuesday

4 Wednesday
5 Thursday
6 Friday
7 Saturday

-e a | A | b | e | r
Specifies the type of event that causes the action being defined to run:

a Specifies an event. This is the default.

A Specifies any type of event (event, error event, or rearm event).

b Specifies an event and a rearm event.

e Specifies an error event.

r Specifies a rearm event.

More than one event type can be specified, for example: **-e ae**. The **-e** flag cannot be specified with the **-b** flag.

-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.

-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.

-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.

-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.

If you are using **mkresponse** on the management server and you want the response to be defined on the management server, do *not* specify the **-p** flag.

-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.

-s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the **logevent**, **notifyevent**, and **wallevent** commands for descriptions of the predefined response scripts provided with the application.

-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.

-u Specifies that the action is to be run when a monitored resource becomes undefined.

-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 *Administering RSCT* 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_IP_AUTHENT

When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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 anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command:

   ```
    mkresponse -n "E-mail root" -d 1+7  \
   -s "/opt/rsct/bin/notifyevent root" -e b   \
   "E-mail root anytime"
   ```

3. To define a response with the name "E-mail root anytime" 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 **/opt/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 "/opt/rsct/bin/notifyevent root" -e a  \
   "E-mail root anytime"
   ```

4. To define a response with the name "E-mail root anytime" that has an action named "E-mail root" to be used any time Saturday and Sunday, that uses the command **/opt/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 "/opt/rsct/bin/notifyevent root" -e b \
   -E LANG="en_US"  "E-mail root anytime"
   ```

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 **/opt/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 "/opt/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"
   ```

7. To define a batching-capable response called "Batched Event Response" without an action script, enter:

   ```
   mkresponse -b "Batched Event Response"
   ```

These examples apply to management domains:

1. To define a response on the management server with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/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 "/opt/rsct/bin/notifyevent root" -e b   \
"E-mail root anytime"
```

2. To define a response on the managed node **nodeB** with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/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 "/opt/rsct/bin/notifyevent root" -e b   \
-p nodeB "E-mail root anytime"
```

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 anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/opt/rsct/bin/notifyevent root** for both events and rearm events, run this command from any node in the domain:

```
mkresponse -n "E-mail root" -d 1+7  \
-s "/opt/rsct/bin/notifyevent root" -e b   \
"E-mail root anytime"
```

2. To define a response on the node **nodeB** in the domain with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday, that uses the command **/opt/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 "/opt/rsct/bin/notifyevent root" -e b   \
-p nodeB -E 'ENV1="PAGE ALL", ENV2="TIMER SET"' \
"E-mail root anytime"
```

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

/opt/rsct/bin/mkresponse

---

mkrole Command

Purpose

Creates new roles.

Syntax

mkrole [**-R** *load_module*] *Attribute*=*Value* [*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 System Management Interface Tool (SMIT) **smit mkrole** fast path to run this command.

If the system is configured to use multiple domains for the role database, the new role is created in the first domain specified by the **secorder** attribute of the roles stanza in the **/etc/nscontrol.conf** file. Use the **-R** flag to create a role in a specific domain.

Every role must have a unique role ID that is used for security decisions. If the **id** attribute is not specified when a role is created, the **mkrole** command automatically assigns a unique ID to the role.

When the system is operating in enhanced (RBAC) mode, roles created in the role database can be immediately assigned to users but are not used for security considerations until the database is sent to the kernel security tables using the **setkst** command.

Flags

Item	Description
-R *load_module*	Specifies the loadable module to use for role creation.

Parameters

Item	Description
Attribute=*Value*	Initializes a role attribute. Refer to the **chrole** command for the valid attributes and values.
Names	Specifies a unique role name string. **Restrictions on Creating Role Names** The *Name* parameter that you specify must be unique, and can be a maximum of 63 single-byte printable characters. 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) • " (quotation mark) • # (pound sign) • , (comma) • = (equal sign) • \ (backslash) • / (forward slash) • ? (question mark) • ' (single quotation mark) • ` (back quotation mark) **Restriction:** The *Name* parameter cannot contain any space, tab, or newline characters.

Security

The **mkrole** command is a privileged command. You must assume a role that has the following authorization to run the command successfully.

Item	Description
aix.security.role.create	Required to run the command.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files Accessed:

Mode	File
rw	/etc/security/roles
r	/etc/security/user.roles

Auditing Events:

Event	Information
ROLE_Create	role

Examples

1. To create the `ManageRoles` role and have the command automatically generate a role ID, use the following command:

   ```
   mkrole authorizations=aix.security.role ManageRoles
   ```

2. To create the `ManageRoles` role in LDAP, use the following command:

   ```
   mkrole -R LDAP authorizations=aix.security.role manageRoles
   ```

Files

Item	Description
/etc/security/roles	Contains the attributes of roles.
/etc/security/user.roles	Contains the role attribute of users.

Related reference:

"mkuser Command" on page 839

Related information:

chrole command

setkst command

Security

mkrpdomain Command

Purpose

Creates a peer domain definition.

Syntax

To create a peer domain definition, by:

- Specifying node names on the command line:

  mkrpdomain [**-t** *TS_port*] [**-g** *GS_port*] [**-Q** *quorum_type* | *quorum_type_name*] [**-c**] [**-m** *fanout*] [**-S** *mode*] [**-k** *cssk_type* [**-r** *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [**-h**] [**-TV**] *peer_domain node_name1* [*node_name2* ...]

- Using a list of node names in an input file:
  mkrpdomain -f | **-F** { *file_name* | **"–"** } [**-t** *TS_port*] [**-g** *GS_port*] [**-Q** {*quorum_type* | *quorum_type_name*}] [**-c**] [**-m** *fanout*] [**-S** *mode*] [**-k** *cssk_type* [**-r** *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [**-h**] [**-TV**] *peer_domain*

To create a peer domain definition with the policy information:

mkrpdomain [**-p** *Policy*] *ApplDomain nameA* [*@host_nameA*] [*nameB* [*@host_nameB*] ...]

Description

The **mkrpdomain** command creates a peer domain definition with the name specified by the *peer_domain* parameter. The nodes that are specified by *node_name* are defined to the new peer domain. A peer domain can be used to provide high-availability services when you configure 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 are successfully defined to the peer domain.

The **mkrpdomain** command fails if one or more of these situations occurs:

- The name of the peer domain is already in use.
- One or more nodes cannot be successfully defined to the peer domain.
- 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.

Use the **-c** flag to enable **mkrpdomain** to continue when there is an error on one of the nodes. The peer domain quorum rules can be modified by using the **-Q** flag. The quorum rules determine under what conditions operational changes, such as starting or stopping resources, and configuration changes, such as adding or removing a node, can be made. Start up quorum defines how many nodes are contacted to get configuration information to start the peer domain. In a typical environment, two quorum rule types are used: normal and quick. For the quick quorum type, only one node is contacted before the peer domain group is started. Operational and configuration quorum rules are the same. To see what quorum rule types are available on a node, run:

```
lsrsrc -c IBM.PeerDomain AvailableQuorumTypes
```

You can use the **-k** flag to set the cluster shared secret key (CSSK). The CSSK is used for message authentication in the peer domain. By default, the CSSK is disabled (that is, set to CSSKTYPE_None). To enable message authentication, use a CSSK value such as CSSKTYPE_DES_MD5 with the **-k** flag. Enabling message authentication affects performance. The complexity of the encryption algorithm determines the effect.

Message authentication also requires that the time-of-day clocks (TODs) of the nodes in the peer domain are synchronized — according to the system time — to within 2 minutes of each other. When the nodes' TODs are synchronized across the peer domain, this function helps to defend against message replay attacks. If the nodes' TODs are not synchronized to within 2 minutes of each other, messages that are passed between a sending node and a receiving node with a time difference that is longer than 2 minutes are discarded.

When message authentication is enabled by using the **-k** flag, a key refresh interval can be specified by using the **-r** flag. By default, the key is refreshed daily.

To change the CSSK type for a peer domain, use the **chrsrc** command. For example:

```
chrsrc -c IBM.RSCTParameters CSSKType=cssk_type
```

To list the CSSK type that is used for an online peer domain, use the **lsrsrc** command. For example:

```
lsrsrc -c IBM.RSCTParameters CSSKType
```

To cause the CSSK to be refreshed, use the **runact** command. For example:

```
runact -c IBM.PeerDomain UpdateKey
```

For information about setting up and managing CSSK settings, see the *Administering RSCT* guide.

Use the **-6** flag to establish a peer domain in which the IPv6 addresses that are configured on the nodes' network interfaces are visible as resources in **IBM.NetworkInterface** class. These IPv6 addresses are not used for heartbeating or internal peer domain operations. If the -6 flag is not specified, no IPv6 addresses are visible as resources in **IBM.NetworkInterface**.

The **mkrpdomain** command does not bring the peer domain online automatically. To bring the peer domain online, run the **startrpdomain** command. You can add nodes to the peer domain by using the **addrpnode** command. To remove nodes from the peer domain, use the **rmrpnode** command.

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

Item	Description
-6	Specifies that the IPv6Support persistent class attribute of the **IBM.NetworkInterface** class has a value of 1 rather than the default (0) in the peer domain that is to be created. For any IP interface on any node in a cluster that has one or more IPv6 addresses configured, only one of these IPv6 addresses are made visible as a resource in **IBM.NetworkInterface**. Therefore, if a network interface has IPv4 addresses and IPv6 addresses configured on it, two resources in **IBM.NetworkInterface** refers to the interface (through the Name attribute), one with the IP address value set to the primary IPv4 address, and one with the selected IPv6 address. If multiple IPv6 addresses are configured on an interface, preference is given to global addresses over link-local addresses for representation as a resource. In addition, IPv6 addresses are used for heartbeating or internal peer domain operations. **Note:** Even if IPv6Support is changed, the current registered applications do not receive the notification for any resource addition or deletion until the domain or the **IBM.ConfigRM** class is restarted.
-c	Continues to run the **mkrpdomain** command on the remaining nodes. By default, if the **mkrpdomain** command fails on any node, it fails on all nodes. The **-c** flag overrides this behavior, so that the **mkrpdomain** command runs on the other nodes, even if it fails on one node.
-C *cluster_type*	Specifies the cluster type. Valid values are as follows: **0** Creates a peer domain. This value is the default. **1** Creates a peer domain and the underlying Cluster-Aware AIX (CAA) cluster. If you specify the **-C 1** flag, you must also specify a repository disk by using the **-R** flag. Also, you can optionally specify one or more shared disks by using the **-D** flag.
-D *shared_disk1* [,*shared_disk2*...]	Specifies one or more shared disks for a CAA cluster. If you specify the **-D** flag, you must also specify the **-C** and **-R** flags.

Item

-f | **-F** { *file_name* | **"-"**

Description

Specifies that node names are read from a file or from standard input. Use **-f** *node_file* or **-F** *node_file* to read the node names from a file.

Note: The command requires that the following conditions be met to display a valid output:

- Specify 1 node name per line. The command ignores any blank characters to the left of the node name.
- Use a number sign (#) to indicate that the remainder of the line (or the entire line if the # is in column 1) is a comment.
- Specify the actual host name of the node by using @ sign without any space between node name and its host name. An example of the syntax follows:

  ```
  [nodeA@hostA]
  ```

By default, all of the nodes that are listed in *node_file*:

- are group services group leader candidates
- are used for quorum decisions
- have access to the peer domain's tiebreaker mechanism

You can customize node characteristics by using an at sign (@) control character followed by one or more of these special characters:

P | p Specifies that the node is a group services group leader candidate.

Q | q Specifies that the node is a quorum node.

B | b Specifies that the node has access to the peer domain's tiebreaker mechanism. B or b can be specified for quorum nodes only.

! Specifies that the node does not have a certain characteristic. For example, **!Q** indicates that the node is not a quorum node.

When customizing node characteristics, consider the following (where x is P, Q, or B):

- Use only one @ control character per line, followed immediately by one or more special characters, after the node name and before any comments.
- Do not specify !QB for a node, as it results in an error.
- If you use a node number, add it after the node name and before any comments. The node number can precede or follow the node characteristic specifications.
- If x is specified for one or more nodes and !x is not specified for any nodes, the nodes that do not have an x specified are assumed to have a value of !x.
- If !x is specified for one or more nodes and x is not specified for any nodes, the nodes that do not have an !x specified are assumed to have a value of x.
- If x and !x are specified for different nodes in the same node file, all of the nodes in the file must have a specification of x or !x.

See the *Administering RSCT* for more information.

Use **-f "-"** or **-F "-"** to read the node names from standard input.

-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 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12348.

-h

Writes the command's usage statement to standard output.

Item	Description
-k *cssk_type*	Specifies the cluster shared secret key (CSSK) to be used for message authentication in the peer domain. Use the CSSK that best suits your applications in terms of the degree of data protection, overhead, and performance. The longer the key and message digest, the stronger the encryption algorithm. The stronger the algorithm, the slower the performance. The valid key types are as follows: **CSSKTYPE_None** Indicates that message authentication is disabled. This is the default value. **Note:** If the **-S** flag is specified with mode value `nist_sp800_131a`, the default CSSK type is `CSSKTYPE_AES256_SHA256`. **CSSKTYPE_DES_MD5** Indicates that a Data Encryption Standard (DES) key with the message digest function MD5 is used to generate a 16-byte signature. This CSSK is recommended if a high degree of data protection is not required and if you want good performance with less data overhead. **CSSKTYPE_3DES_MD5** Indicates that a triple DES key with an MD5 digest is used to generate a 16-byte signature. Compared to **CSSKTYPE_DES_MD5**, this CSSK provides added data protection with slower performance, but with the same data overhead. **CSSKTYPE_AES256_MD5** Indicates that an Advanced Encryption Standard (AES) 256-bit key with an MD5 digest is used to generate a 24-bit signature. This CSSK provides more data protection than **CSSKTYPE_3DES_MD5**, but with slower performance and more data overhead. The following CSSK types are compliant with the National Institute of Standards and Technology (NIST) Special Publications SP800-131a. You must be running RSCT 3.2.0.0, or later, to configure these key types. **CSSKTYPE_AES128_SHA256** Indicates that an Advanced Encryption Standard (AES) 128-bit key that has an SHA-1 (Secure Hash Algorithm) 256-bit digest is used to generate a 16-byte signature. **CSSKTYPE_AES128_SHA512** Indicates that an AES 128-bit key that has an SHA-1 512-bit digest is used to generate a 16-byte signature. **CSSKTYPE_AES256_SHA256** Indicates that an AES 256-bit key that has an SHA-2 256-bit digest is used to generate a 32-byte signature. **CSSKTYPE_AES256_SHA512** Indicates that an AES 256-bit key that has an SHA-2 512-bit digest is used to generate a 32-byte signature. **Notes:** • You must be running RSCT 2.4.7.1 or later to use this flag. • If the **-S** flag is specified with the mode value `nist_sp800_131a`, the CSSK type must be either `CSSKType_None` or a key type that is compliant with the mode. If the created domain is compliant with the mode value `nist_sp800_131a`, and the **-k** flag is not specified, the domain is configured to use CSSK type `CSSK_AES256_SHA256`.

Item	Description
-m *fanout*	Specifies the maximum number of threads to use in parallel operations for the specified peer domain. This value is stored as a persistent attribute in the peer domain's **IBM.PeerNode** class. *fanout* can be an integer from **16** to **2048**. If this flag is not specified, the default value (**128**) is used.
-p *Policy*	Reads the policy from the user input when the **mkrpdomain** command creates the domain. You can use this command to specify the policy information when you create the domain. The valid values for the *Policy* attribute are 0 and 1. If you do not specify the **-p** flag for the **mkrpdomain** command, the default value 0 is set in non-CAA clusters and 1 is set in CAA clusters. If the value of policy is set as 1, the **Name** field of the **IBM.PeerNode** class is maintained in sync with the host name of the **IBM.PeerNode** class. If the value of policy is set as 0, the **Name** field is not maintained in sync with the host name, irrespective of the domain. However, the **-p** *0* flag cannot be specified for CAA domain as a limitation. The policy information can be changed by using a **chrsrc** class action after the cluster is created.
-Q *quorum_type* | *quorum_type_name*	Specifies the quorum rules that are used for startup, operational, and configuration quorum. Startup quorum defines how many nodes are contacted to obtain configuration information before the peer domain is started. Operational quorum defines how many nodes must be online to start and stop resources and how tie breaking is used. Configuration quorum defines how many nodes must be online to change the peer domain (adding or removing a node, for example). To see what quorum rule types are available on a node, run: `lsrsrc -c IBM.PeerDomain AvailableQuorumTypes` The valid values are as follows: **0 | normal** Specifies normal quorum rules. This value is the default. For startup quorum, at least half of the nodes are contacted for configuration information. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum. **1 | quick** Specifies quick quorum rules. For startup quorum, even if no other nodes can be contacted, the node still comes online. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum.

Item	Description
-r *refresh_interval*	Specifies the CSSK refresh interval when message authentication is enabled in the peer domain. This is the interval at which the CSSK is refreshed. The format of *refresh_interval* is: *dd***:***hh***:***mm***:***ss*, where *dd* is the number of days between key refreshes, *hh* is the number of hours, *mm* is the number of minutes, and *ss* is the number of seconds. The *refresh_interval* value can be truncated on the right, so **-r 5** means refresh every 5 days and **-r 0:12** means refresh every 12 hours. The default refresh interval is 1 day. The minimum refresh interval is 30 seconds. The maximum refresh interval is 30 days. The **-r** flag can be specified when the **-k** flag is used. You must be running RSCT 2.4.7.1 or later to use this flag.
-R *repository_disk*	Specifies the repository disk for a CAA cluster. If you specify the **-R** flag, you must also specify the **-C** flag.
-S *mode*	Enforces a security compliance mode for RSCT in the peer domain. The *mode* parameter can have the following values: **none** The domain does not enforce a security compliance mode. **nist_sp800_131a** RSCT is configured to be compliant with the National Institute of Standards and Technology (NIST) Special Publications SP800-131a. This mode value requires that all nodes that are specified in the **mkrpdomain** command must already be migrated to `nist_sp800_131a` mode or the nodes must be configured to use public or private keys that are compliant with this specification. **Note:** You must be running RSCT 3.2.0.0, or later, to use the **-S** flag.
-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 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12347.
-T	Writes the command's trace messages to standard error. For your software service organization's use only.
-V	Writes the command's verbose messages to standard output.

Parameters

peer_domain
Specifies the name of the new peer domain to be created. You can use these ASCII characters only in the peer domain name: **A** to **Z**, **a** to **z**, **0** to **9**, **.** (period), and _ (underscore). In addition, the peer domain 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 host name. The node name must resolve to an IP address.

Security

The user of the **mkrpdomain** command requires **write** permission to the **IBM.PeerDomain** resource class on each node that is to be defined to the peer domain. This permission 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 runs **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.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

Any node to be defined to the peer domain must be reachable from the node on which this command runs.

Implementation Specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) file set for AIX.

Standard Input

When the **-f "-"** or **-F "-"** flag is specified, this command reads one or more node names from standard input.

Standard Output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To define a peer domain that is called **ApplDomain** that consists of a node that is called **nodeA**, run this command on **nodeA**:

```
mkrpdomain ApplDomain nodeA
```

2. To define a peer domain that is called **ApplDomain** that consists of three nodes that are called **nodeA**, **nodeB**, and **nodeC**, run this command on **nodeA**, **nodeB**, or **nodeC**:

   ```
   mkrpdomain ApplDomain nodeA nodeB nodeC
   ```

3. To define a peer domain that is called **ApplDomain** that consists of 2 nodes that are called **nodeA** and **nodeB**, with a topology services port number of 1200 and a group services port number of 2400, run this command on **nodeA** or **nodeB**:

   ```
   mkrpdomain -t 1200 -g 2400 ApplDomain nodeA nodeB
   ```

4. To define a peer domain that is called **ApplDomain** that consists of 2 nodes that are called **nodeA** and **nodeB** by using message authentication key algorithm **CSSKTYPE_DES_MD5**, run this command on **nodeA** or **nodeB**:

   ```
   mkrpdomain -k CSSKTYPE_DES_MD5 ApplDomain nodeA nodeB
   ```

5. To define a peer domain that is called **ApplDomain** that consists of the nodes **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, by using the **/pd/pdnodes.config** file, run the following command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of **/pd/pdnodes.config** are as follows:

   ```
   # peer domain nodes for mkrpdomain
   nodeA       # dev node
   nodeB       # dev node
   nodeC       # prod node
   nodeD       # test node
   nodeE       # test node
   ```

6. To define a peer domain that is called **ApplDomain** that consists of **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, by using the **/pd/pdnodes.config** file, which specifies that **nodeA** has access to the peer domain's tiebreaker mechanism, **nodeB** and **nodeC** cannot be used in quorum decisions, and **nodeC** and **nodeD** cannot be the group services group leader, run the following command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of **/pd/pdnodes.config** are as follows:

   ```
   # peer domain nodes for mkrpdomain
   nodeA    @QB     # dev node
   nodeB    @!Q     # dev node
   nodeC    @!Q!P   # prod node
   nodeD    @!P     # test node
   nodeE    @Q      # test node
   ```

7. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, with the policy **NamePolicy 1**, run the following command:

   ```
   mkrpdomain -p 1 ApplDomain nodeA nodeB
   ```

   NamePolicy 1 means that any change in host name also updates the node name. In this case, the host name is not specified in the beginning. Hence, the node names (**nodeA** and **nodeB**) are set as host names for the respective nodes.

8. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, whose host names are **hostA** and **hostB**, run the following command:

   ```
   mkrpdomain ApplDomain nodeA@hostA nodeB@hostB
   ```

   These host names are the actual host names that are used for communication.

Location

/opt/rsct/bin/mkrpdomain

Files

The **/etc/services** file is modified.

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 or exclusive rset (xrset) 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. When used to create an xrset, the **mkrset** command changes the state of the corresponding CPUs on the system to exclusive mode. Creating an xrset requires root privilege.

Flags

Item	Description
-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

Item	Description
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 and CAP_PROPAGATE capability.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/bin/mkrset	Contains the **mkrset** command.

Related reference:
"lsrset Command" on page 506
Related information:
attachrset command
detachrset command
execrset command

mkrsrc Command

Purpose

Defines a new resource.

Syntax

To define a new resource, using data that is...

- entered on the command line:
  mkrsrc [**-a** | **-N** { *node_file* | **"-"** }] [**-v**] [**-h**] [**-TV**] *resource_class attr=value...*
- predefined in an input file:
  mkrsrc -f *resource_data_input_file* [**-v**] [**-a** | **-N** { *node_file* | **"-"** }] [**-h**] [**-TV**] *resource_class*

To display the names and datatypes of the command arguments:

mkrsrc -l [**-h**] *resource_class*

To see examples of the **mkrsrc** 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 **reqd_for_define** (required) or **option_for_define** (optional). Only attributes that are designated as **reqd_for_define** or **option_for_define** can be defined using the **mkrsrc** command. The **lsrsrcdef** command also identifies the datatype for each attribute. The value specified for each attribute must match this datatype.

To verify that all of the attribute names that are specified on the command line or in *resource_data_input_file* are defined as persistent attributes and are designated as **reqd_for_define** or **option_for_define**, use the **-v** flag. When the **mkrsrc** command is run with the **-v** flag, the resource is not defined. Instead, the resource attributes are merely verified to be persistent and designated as **reqd_for_define** or **option_for_define**. Once you have run **mkrsrc -v** to verify that all of the attributes that are specified on the command line or in *resource_data_input_file* are valid, you can issue the **mkrsrc** command without the **-v** flag to define the new resource.

If you are running in an RSCT peer domain or on the management server in an RSCT management domain and the resource class management type is subdivided, you can create the same resource on multiple nodes in one of two ways. The first way is to use the **-N** *node_file* flag to indicate that the node names to create the resources on are in a file. Use **-N "-"** to read the node names from standard input. The second way is to specify multiple node names in the **NodeNameList** resource attribute. The **NodeNameList** attribute defines where the resource is created when a cluster is present. If the **NodeNameList** attribute is not used, the resource is created on the local node. To find out if a resource class management type is subdivided, enter **lsrsrcdef -c** *resource_class* | **grep properties**.

Flags

-e Displays examples of **mkrsrc** command-line input for:

1. required attributes only
2. required and optional attributes

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information.

-l Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the define request. Use this flag to list any defined command arguments and the datatypes of the command argument values.

-N { *node_file* **| "-" }**
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "-"** to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If the resource class management type of the resource that is to be defined is subdivided and **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-v Verifies that all of the attribute names specified on the command line or in the input file are defined as persistent attributes and are designated as **reqd_for_define** or **option_for_define**. The **mkrsrc** command does *not* define any resources when you use this flag.

-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 **reqd_for_define** or **option_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 *Administering RSCT* guide for information about the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect 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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Implementation Specifics

This command is part of the **rsct.rmc** fileset for the AIX® operating system.

Standard Output

- All command output is written to standard output.
- When the **-h** flag is specified, this command's usage statement is written to standard output.
- When the **-V** flag is specified, this command's verbose messages (if there are any available) 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 attributes (**Binary**, **SD**, **StringArray**, and **SDArray**) 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}]}'
   ```

5. To create resources for the IBM.Example class on multiple nodes in a peer domain, run this command:

   ```
   mkrsrc -N /u/joe/common_node_file  IBM.Example  Name=Example_bar1 \
   Binary="0xaabbccddeeff00"
   ```

   where the contents of **/u/joe/common_node_file** look like this:

   ```
   # common node file
   #
   node1.ibm.com      main node
   ```

```
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

6. To create resources of the IBM.Example class on multiple managed nodes in a management domain, run this command on the management server:

```
mkrsrc IBM.Example  Name=Example_bar1 Binary="0xaabbccddeeff00" \
NodeNameList='{"mgnode1.ibm.com","mgnode2.ibm.com"}'
```

where the contents of **/u/joe/common_node_file** look like this:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

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

/opt/rsct/bin/mkrsrc

---

mkrtc Command

Purpose

Configures or unconfigures Power SC real-time compliance for the operating system instance.

Syntax

To configure Power SC real-time compliance:

mkrtc -e *email1, email2...* [**-a** *alertStyle*] [**-d** *debug*] [**-i** *infoLevel*] [**-s** *emailSubject*] [**-c** *minCheckTime*]

To unconfigure Power SC real-time compliance:

mkrtc -u

Description

The **mkrtc** command is used to configure or unconfigure Power SC real-time compliance. To configure the **-e** flag, the email addresses must be provided as arguments. All other flags are optional. The **mkrtc** command saves the options to the `/etc/security/rtc/rtcd.conf` file, adds the Power SC real-time compliance entry to the `/etc/inittab`, and starts the **rtcd** daemon.

On unconfiguration, the **mkrtc** command removes the entry from the `/etc/inittab` file and stops the **rtcd** daemon.

Flags

Flag	Description
-a *alertStyle*	Specifies the alert style. The following are valid values: • `Once`: Alerts once for the same set of compliance violations. This is the default alert style. • `Event`: Alerts once for the same set of compliance violations, but keeps alerting for each file modification event. • `Always`: Alerts compliance violations and the file modification. It keeps alerting for the file modification.
-c *minCheckTime*	Specifies the minimum amount of time between the compliance verifications. This flag checks the Power SC for compliance regularly even without file modification triggers, so that the **mkrtc** command can detect compliance implications in the files that are created by the user. For example, this flag can detect the `.rhost` file creation in the home directory that can have compliance implication. The default minimum time is 30 minutes. If this value is set to 0, it indicates that the compliance check is never run unless the files are modified.
-d *debug*	Specifies the debug option to be turned on or off. The valid values are `On` or `Off`. The default value is `Off`.
-e *email1, email2...*	Provides a comma-separated list of emails to which email alerts are to be sent.
-i *infoLevel*	Specifies the information level of file modification events.
-s *emailSubject*	Provides the subject line to be used for the email alert.
-u	Unconfigures the Power SC real-time compliance.

On configuring Power SC real-time compliance, the **mkrtc** command performs the following tasks:

1. Updates the `/etc/security/rtc/rtcd.conf` file with the options from the command line.
2. Updates the `/etc/inittab` file with *pscrtc:2:wait: /usr/bin/startsrc -s rtcd*.
3. Starts the **rtcd** daemon.

On unconfiguration, the **mkrtc** command performs the following tasks:

1. Removes the Power SC real-time compliance entry from `/etc/inittab` file.
2. Stops the **rtcd** daemon.

Security

Only the root user and users with **aix.security.aixpert** authorization are authorized to run this command.

Exit Status

Value	Description
0	The command runs successfully.
>0	An error occurred. The printed error message lists further details about the type of failure.

Examples

1. To configure Power SC real-time compliance, type the following command:

   ```
   # mkrtc -e test@abc.com,dummy@abc.com -a event
   ```

   This command configures Power SC real-time compliance to send compliance violation alert and file modification events to *test@abc.com* and *dummy@abc.com*. The alert style is set to *event*.
2. To unconfigure Power SC real-time compliance, type the following command:

   ```
   # mkrtc -u
   ```

Files

Mode	File
rw	/etc/security/rtc/rtcd.conf

mkseckrb5 Command

Purpose

Migrates existing operating system users to Kerberos.

Syntax

mkseckrb5 [**-h** | [**-r**] [**user_name...**]]

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.

Item	Description
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

Item	Description
-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.

Item	Description
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

Item	Description
/usr/sbin/mkseckrb5	Contains the **mkseckrb5** command.

mksecldap Command

Purpose

Sets up an AIX system as an LDAP server or client 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*] [**-x** *proxyDN* **-X** *proxypasswd*] [**-u** *NONE*] [**-v** *LDAPVersion*] [**-U**] [**-j** <*ssl* | *tls* | ssl*tls* | *none* | *sslonly*>]

The syntax to set up a client is:

mksecldap -c -h *serverlist* **-a** *bindDN* **-p** *bindpwd* [**-d** *baseDN*] [**-n** *serverport*] [**-k** *SSLkeypath*] [**-w** *SSLkeypasswd*] [**-t** *cachetimeout*] [**-C** *cachesize*] [**-P** *NumberofThreads*] [**-T** *heartBeatInt*] [**-M** *searchMode*] [**-D** *defaultEntry*] [**-A** *authType*] [**-i** *databaseModule*] [**-u** *userlist*] [**-U**] [**-j** <*ssl* | *tls*>]

Description

The **mksecldap** command can be used to set up IBM Directory servers and clients for security authentication and data management.

Notes:

1. The client (**-c** flag) and the server (**-s** flag) options cannot be used at the same time. When setting up a server, the **mksecldap** command might need to be run twice on that machine. Once to set up the server, and again to set up the system as a client.
2. The name and location of the LDAP server configuration file depends on the version of LDAP software installed. Refer to the LDAP software documentation of the installed release for more information.

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 pre-configuration 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.

   Note: If IBM Directory Server 5.1 or later is in use then prompts for the ldapdb2 password will appear.
2. If IBM Directory Server 6.0 or later is being configured then an LDAP server instance with the default name of `ldapdb2` is created. A prompt is displayed for the encryption seed to use to create the key stash files. The input encryption seed must be at least 12 characters.
3. 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.

4. Create the base DN (suffix) of the directory information tree (DIT). It is required that the base DN start with one of these attributes: **dc**, **o**, **ou**, **c**, **cn**. If no baseDN is supplied from the command line, the default suffix is set to **cn=aixdata** and the user/group data is placed under the **cn=aixdata** DN. Otherwise, the **mksecldap** command uses the user-supplied DN specified with the **-d** option. Users and groups will be exported to LDAP using the **sectoldif** command. The directory information tree (DIT) that will be created by default is shown below.

```
      <user supplied suffix>
                |
    --------------------------
    |                        |
  ou=People                ou=Groups
```

5. If **-u** *NONE* is not specified, then export the data from the security database files from the local host into the LDAP database. If **-u** *NONE is* specified, then **mksecldap** does not create the `ou=People` and `ou=Group` containers as it normally would, nor does it export users and groups. Depending on the **-S** option, the **mksecldap** command exports 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).
6. Set the LDAP server administrator DN and password.
7. Set the server to listen to a specified port if the **-n** option is used. The default port is 389. Also, TLS use this port as default port (636 for SSL).
8. Updates the **/usr/lib/security/methods.cfg** file with the LDAP module configuration. If the **-i** option is entered from the command line, it also sets a LDAPA authentication-only module and a compound loadmodule (for example, LDAPAfiles when the **-i** *files* option is specified) with LDAPA serves for authentication and the databaseModule serves for identification.
9. Create the proxy entry if the -x and -X options are specified. Create an ACL for the base DN using the proxy entry. The default ACL can be found in /etc/security/ldap/proxyuser.ldif.template. The proxy entry can be used by client systems to bind to the server (see client setup section in this file).
10. Set the server to use SSL (secure socket layer) or TLS (transport layer security) if the **-k** option is specified for secure data transfer between this server and the clients. This setup requires the **GSKIT** to be installed and creation of an SSL or TLS key.
11. Installs the **/usr/ccs/lib/libsecldapaudit.a** LDAP server plug-in. This plug-in supports AIX audit of the LDAP server.
12. Start/restart the LDAP server after all the above is done.
13. Add the LDAP server process (**slapd**) to **/etc/inittab** to have the LDAP server start after reboot.

Note: The **-U** option resets a previous setup for the server configuration file. It has no effect on the database. The first time the **mksecldap** command is run, it saves two copies of the server configuration file in the **/etc/security/ldap** directory. One is saved as the server configuration file name appended with **.save.orig** and the other is appended with **.save**. During each subsequent run of the **mksecldap** command, only the current server configuration is saved as a **.save** file. The undo option restores the server configuration file with the **.save** copy. In AIX 5.3 it is possible to invoke **mksecldap -s** in succession to create and populate multiple suffixes. If this has been performed then the **.save.orig** file will need to be manually restored in order to revert to the initial configuration file.

Client Setup

Make sure that the LDAP client fileset is installed and the LDAP server has been setup and is running. The **mksecldap** command performs the following steps during 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, and sets up the base DNs accordingly. If the server has multiple user or group bases, you must supply the **-d** option with a Relative Distinguished Name (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 the following entities from the server and save any that are found:

- hosts
- networks
- services
- netgroups
- protocols
- rpc
- authorizations
- roles
- privcmds
- privdevs
- privfiles
- usrkeystore
- grpkeystore
- efscookies
- admkeystore
- domains
- domobjs

3. Determines the schema type used by the LDAP server - **AIX** specific schema, **RFC 2307** schema, **RFC 2307** schema with full AIX support, or Microsoft Services for UNIX 3.0 schema. 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 or TLS for secure data transfer between this host and the LDAP server. This step requires that the client SSL or TLS key and the key password are created in advance, and the server must be setup to use SSL or TLS for the client SSL or TLS to work.
5. Encrypts the bind password.
6. Saves the LDAP server bind DN and password. The DN/password pair must exist on the LDAP server. If the bind DN and password are not given, **mksecldap** uses anonymous bind. Some of the data might not be returned from the LDAP server with anonymous bind. Consult your LDAP administrator before you choose anonymous bind.
7. Sets the optionally specified configuration values as defined in the client setup flags section.
8. 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, see the following note.
9. Starts the client daemon process (**secldapclntd**).
10. Adds the client side daemon process to **/etc/inittab** to have this daemon start after a reboot.

Note: All client configuration data is saved to the **/etc/security/ldap/ldap.cfg** configuration file. The **-U** option resets a previous setup to the **/etc/security/ldap/ldap.cfg** file by replacing the file with the configuration stored in **/etc/security/ldap/ldap.cfg.save**. 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

Item	Description
-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**.
-j < *ssl* | *tls* | ssl*tls* | *none* | *sslonly*>]	Specifics the encryption connection type that is used during the communication with the LDAP clients. Valid values are SSL, TLS, SSLTLS, and SSLONLY. If the **-k** and **-w** flags are specified without the **-j** flag, the default connection type is SSL.
-k *SSLkeypath*	Specifies the full path to the SSL or TLS key database of the server.
-n *port*	Specifies 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 or TLS key.
-U	Specifies to undo the previous server setup to the LDAP configuration file. The database is not affected.
-u `NONE`	Specifies not to migrate users and groups from local system. The only valid value is `NONE`. Any other values are ignored. When this option is used, **mkseccldap** does not create the `ou=People` and `ou=Group` containers as it normally would, nor does it export users and groups. No **-S** option is required with this option.
-v *LDAPVersion*	Denotes a specific version of the LDAP server fileset to configure. The value must be in the format #.# where # is a number. For example, `6.0`. If not specified, the **mkseccldap** command configures the most recent version of the LDAP server fileset that is installed.
-X *proxypasswd*	Specifies the password for the proxy DN.
-x *proxyDN*	Specifies the DN of the proxy entry. This entry can be used by client systems to bind to this server.

For Client Setup

Item	Description
-a *bindDN*	Specifies the DN to bind to the LDAP server. The DN must exist on the LDAP server. If authtype is unix_auth, bindDN must have read access to the userPassword field on the LDAP server. Without the -a option, **mkseccldap** configures anonymous bind. **Note:** Some of the data might not be retrieved from the LDAP server with anonymous bind. Consult your LDAP server administrator about using anonymous bind.
-A *authType*	Specifies the authentication mechanism used to authenticate users. Valid values are **unix_auth** and **ldap_auth**. The default is **unix_auth**. The values are defined as follows: • **unix_auth** - Retrieve user password from LDAP and perform authentication locally. • **ldap_auth** - Bind to LDAP server, sending password in clear text, for authentication. **Note:** When using **ldap_auth** type authentication, the use of SSL or TLS is strongly recommended since during authentication passwords will be sent in clear text to the LDAP server.
-i *databaseModule*	Specifies the configuration of LDAP as the authentication-only module (LDAPA) of a compound loadmodule. The databaseModule option specifies the database module of the compound loadmodule.
-j <*ssl* | *tls*>	Specifics the encryption connection type that is used during the communication with the LDAP server. Valid values are SSL and TLS. If the **-k** and **-w** flags are specified without the **-j** flag, the default connection type is SSL.
-c	Indicates the command is being run to setup the client.

Item	Description
-C *Cachsize*	Specifies the maximum number of user entries that can be used in the client-side daemon cache. Valid value is in the range 100 - 65536 for user cache. The default is 1000. The valid range for the group cache is 10-65536. The default value is 100. If you set the user cache entry in the **start-secldapclntd** command, by using the **-C** option, the group cache is set to 10% of the user cache.
-D *defaultEntryLocation*	Specifies the location of the default entry. Valid values are **ldap** and **local**. The default is **ldap**. The values are defined as follows: • **ldap** - Use the default entry in LDAP for all attribute default values. • **local** - Use the default stanza from local **/etc/security/user** file for all attribute default values.
-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 or TLS key.
-M *searchMode*	Specifies the set of user and group attributes to be retrieved. Valid values are **ALL** and **OS**. The default is **ALL**. The values are defined as follows: • **ALL** - Retrieve all attributes of an entry. • **OS** - Retrieve only the operating system required attributes of an entry. Non-OS attributes like telephone number, binary images etc. will not be returned. **Note:** Use OS only when entries have many non-OS required attributes or attributes with large value, e.g. binary data, to reduce sorting effort by the LDAP server.
-n *serverport*	Specifies the port number that the LDAP server is listening to.
-p *bindpasswd*	Specifies the clear text password for the bindDN used to bind to the LDAP server.
-P *NumberofTreads*	Specifies the number of threads that the client side daemon uses. Valid values are 1-256. The default value is 10.
-t *Cachetimeout*	Specifies the maximum time length that a cache entry expires. Valid values 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 user names to enable for LDAP authentication. These users will have their registry and SYSTEM attributes set to use LDAP. Specify **ALL** to enable all users on the client. **Note:** Alternatively, the SYSTEM attribute in the default stanza of **/etc/security/user** can be set to LDAP, allowing only LDAP users to log in. Setting the SYSTEM attribute to LDAP or compat allows both LDAP users and local users to log in to the system.
-w *SSLkeyfilepath*	Specifies the password for the client SSL or TLS key.
-U	Specifies to undo the previous client setup to the LDAP client configuration file.

Security

A user with the **aix.security.ldap** authorization is authorized to use this command.

Examples

1. To setup a LDAP server of RFC2307AIX specific schema for users and groups, enter:

   ```
   mksecldap -s -a cn=admin -p adminpwd -S rfc2307aix
   ```

   This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is exported from local files to the default **cn=aixdata** suffix using RFC2307AIX schema.
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 exported from local files to the **o=mycompany,c=us** suffix using RFC2307 schema. 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.

3. To setup a LDAP server of RFC2307AIX schema type and create a proxy account, enter:

```
mksecldap -s -a cn=admin -p adminpwd -d c=us -S rfc2307aix -x cn=proxy,c=us -X proxypwd
```

This sets up a LDAP server with LDAP server administrator DN being **cn=admin**, password being **adminpwd**. User and group data is exported from local files to the **c=us** suffix using RFC2307AIX schema. A proxy identity is setup with DN being **cn=proxy,c=us** and password **proxypwd**. The ACL specified in **/etc/security/ldap/proxy.ldif.template** will also have been applied on the server for the **cn=proxy,c=us** DN.

4. To undo a previous server setup:

```
mksecldap -s -U
```

This undoes the previous setup to the server configuration file. Note, for safety reasons, this does not remove any database entries or database created by a previous setup. One has to remove the database entries/database manually if they are not needed any more.

5. 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 administrator DN and password is 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.

6. 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.

7. To setup a client to talk to **server4.ibm.com** and use **ldap_auth** authentication with a proxy bind, enter:

```
mksecldap -c -a cn=proxy,c=us -p proxypwd -h server4.ibm.com -A ldap_auth
```

This sets up an LDAP client to bind to the LDAP server with the **cn=proxy,c=us** DN. Because the administrator DN is not used, the access granted to the client is dependent on the ACL setup on the LDAP server for the **cn=proxy,c=us** DN. The client is also setup to use **ldap_auth**-type authentication which sends passwords in clear text to the LDAP server for comparison.

Note: When using **ldap_auth**-type authentication, the use of SSL or TLS is strongly recommended because during authentication passwords will be sent in clear text to the LDAP server.

8. To undo a previous client setup, enter:

```
mksecldap -c -U
```

This undoes the previous setup to the **/etc/security/ldap/ldap.cfg** file. This does not remove the **SYSTEM=LDAP** and **registry=LDAP** entries from the **/etc/security/user** file.

9. To setup a client using LDAP as authentication-only module, and using files for user identification, enter:

```
mksecldap -c -a cn=admin -p adminpwd -h server1.ibm.com -i files -A ldap_auth
```

This sets up a LDAPAfiles compound loadmodule, where the module LDAPA is used for user authentication and files is used for user identification. Authentication is set to ldap_auth.

Files Accessed

Item	Description
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:

start-secldapclntd command

flush-secldapclntd command

sectoldif command

/etc/security/ldap/ldap.cfg file

Setting up an LDAP security information server

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

Item	Description
-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 reference:

"keyadd Command" on page 152

"keylist Command" on page 157

"keypasswd Command" on page 160

Related information:

certlist command

mksensor Command

Purpose

Defines a sensor or a microsensor to the resource monitoring and control (RMC) subsystem.

Syntax

To define a sensor:

mksensor [**–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–i** *seconds*] [**–c** *n*] [**–e 0** | **1** | **2**] [**–u** *user-ID*] [**–h**] [**–v** | **–V**] *sensor_name* [**"**]*sensor_command*[**"**]

To define a microsensor:

mksensor -m [**–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–i** *seconds*] [**–h**] [**–v** | **–V**] *microsensor_name microsensor_module* [[**"**]*microsensor_arguments*[**"**]]

Description

The **mksensor** command defines a sensor resource to the resource monitoring and control (RMC) subsystem. A *sensor* is an RMC resource with attributes that you can monitor. You can use the event-response resource manager (ERRM) commands to set up monitoring of the sensor attributes. The response actions defined will run when a monitored sensor event occurs. This enables administrators to extend RMC monitoring capabilities without having to write a resource manager.

For sensors, the *sensor_command* parameter specifies the command or script that the sensor resource manager will run to set (and then later, update) the sensor attribute values. After the sensor attributes have been monitored, the sensor resource manager sets the attribute values. Then, at defined intervals, the sensor resource manager updates these values.

For microsensors, the *microsensor_module* parameter specifies the path name to the loadable module that the microsensor resource manager will call to set (and then later, update) the microsensor attribute values. After the microsensor attributes have been monitored, the microsensor resource manager sets the attribute values. Then, at defined intervals, the microsensor resource manager updates these values. Use the **-m** flag to create a microsensor.

Alternatively, you can use **chsensor** or **refsensor** to update the sensor or microsensor attribute values. The **lssensor** command displays values for sensor or microsensor attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. To remove a sensor or a microsensor, use the **rmsensor** command.

The **mksensor** command runs on any node. To define a sensor or a microsensor on one or more nodes in a management domain or a peer domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

A sensor consists of the following attributes, which can be set using a sensor command: :

Float32
The type float32 attribute for this sensor resource.

Float64
The type float64 attribute for this sensor resource.

Int32 The type int32 attribute for this sensor resource.

Int64 The type int64 attribute for this sensor resource.

Quantum
The type quantum attribute for this sensor resource.

String The type string attribute for this sensor resource.

Uint32
The type uint32 attribute for this sensor resource.

Uint64
The type uint64 attribute for this sensor resource.

A sensor command sets attribute values by sending the values to standard output in a format that the sensor resource manager can parse. The format is *attr=value*. For example, if the sensor command sets the **Int32** attribute to 57, it writes **Int32=57** to standard output. To set more than one attribute value, the sensor command can write multiple *attr=value* pairs to standard output. The *attr=value* pairs can be on one or more lines. If the sensor command output is not in *attr=value* form, it is assumed to be a string and the value is placed in the **String** attribute.

The sensor command runs using the user ID that creates the sensor resource. Once a sensor resource is monitored, the sensor command is run at intervals specified by the **-i** flag, which is expressed in seconds. The default interval is 60 seconds. Specify a value of 0 to indicate that the sensor command is not to run at intervals. In this case, the **refsensor** command is typically used to update the sensor values.

Use the **-e** flag to control how the exit values from *sensor_command* are interpreted. Depending on this setting, when the exit value of the *sensor_command* is considered to be an error, the sensor attributes are not set and information is written to the audit log.

A microsensor consists of the following attributes, which can be set using a microsensor load module:

Float32
The type **float32** attribute for this microsensor resource

Float32Array
The type **float32** array attribute for this microsensor resource

Float64
The type **float64** attribute for this microsensor resource

Float64Array
The type **float64** array attribute for this microsensor resource

Int32 The type **int32** attribute for this microsensor resource

Int32Array
The type **int32** array attribute for this microsensor resource

Int64 The type **int64** attribute for this microsensor resource

Int64Array
The type **int64** array attribute for this microsensor resource

Quantum
The type **quantum** attribute for this microsensor resource.

String The type **string** attribute for this microsensor resource.

StringArray
The type **string** array attribute for this microsensor resource.

UInt32
The type **uint32** attribute for this microsensor resource.

UInt32Array
The type **uint32** array attribute for this microsensor resource.

UInt64
The type **uint64** attribute for this microsensor resource.

UInt64Array
The type **uint64** array attribute for this microsensor resource.

The microsensor resource manager will make calls to the microsensor load module to set the values of the microsensor attributes. See the *Administering RSCT* for information about how to use microsensors.

Flags

–m Specifies that the resource to be defined is a microsensor resource.

–n *host1***[,***host2***...]**
Specifies one or more nodes on which the sensor should be defined. By default, the sensor is defined on the local node. This flag is only appropriate in a management domain or a peer domain.

–N { *node_file* **| "–" }**
Specifies that node names are read from a file or from standard input.

Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (**#**) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "–"** in a management domain or a peer domain to read the node names from standard input.

–i *seconds*
Specifies the interval at which a sensor command is run to update a sensor's attribute values or a microsensor module is run to update a microsensor's attribute valuesyup. *seconds*, which is an integer value, must be greater than or equal to **10**. The default interval is **60** seconds.

The sensor command is run at the specified interval only when the sensor resource is monitored. The microsensor module is run at the specified interval only when the microsensor resource is monitored. If the interval is set to **0**, the sensor command or microsensor module will not run automatically.a

Using this flag is independent of using the **refsensor** command to refresh a sensor.

–c *n* Specifies whether special handling is required for this sensor. *n* can be one of these values:

0 Indicates that no special handling is required. This is the default.

The sensor command will run at the interval that is defined for *sensor_name*. The sensor command will *not* run when monitoring begins or when the **lssensor** command is run.

1 Indicates that the sensor command will run when monitoring begins. The sensor command will also run at the interval that is defined for *sensor_name*. The sensor command will *not* run when the **lssensor** command is run.

Specfiying this value is not recommended, unless you expect the sensor command to run quickly. If the sensor command does not run quickly, it could block other requests to the sensor resource manager. These requests will not be processed until the sensor command finishes running.

2 Indicates that output from the command in the **SavedData** field is not saved permanently to **SavedData** persistent resource attributes. If this value is not specified, the sensor resource manager updates data in the registry's resource table whenever the command's standard output contains the line: **SavedData="***any-string***"**.

3 Indicates a combination of values **1** and **2**.

4 Indicates that the sensor resource manager will run the sensor command after monitoring has stopped.

5 Indicates a combination of values **1** and **4**.

6 Indicates a combination of values **2** and **4**.

7 Indicates a combination of values **1**, **2**, and **4**.

–e 0 | 1 | 2
Specifies how the sensor resource manager interprets the exit values of *sensor_command*, as follows:

0 No exit value from *sensor_command* is an error.

1 An exit value other than **0** from *sensor_command* is an error.

2 An exit value of **0** from *sensor_command* is an error.

The default value is **1**. The sensor attributes are not updated when the exit value is interpreted as an error. For an error, information is written to the audit log.

–u *user-ID*
Specifies the name of a user whose privileges will be used to run the sensor command. The user should already be defined on the system. The default value for *user-ID* is the user name that is associated with the current effective user ID.

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

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

Parameters

["]*microsensor_argument***["]**
Specifies a string that will be passed to the microsensor module callback function. The microsensor resource manager will break the string into an array of strings based on blank characters in the microsensor argument. The microsensor argument cannot be changed once the microsensor is defined.

If the microsensor argument contains any blank characters or any special characters that can be interepreted by the shell, it must be enclosed in double quotation marks. When the microsensor argument is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**).

microsensor_module
Specifies the path name to the loadable microsensor module. A signature for the module is stored by the microsensor resource manager and is verified when the module is used. The microsensor module cannot be changed once the microsensor is defined.

microsensor_name
Specifies the name of the microsensor that is to be defined.

["]*sensor_command***["]**
Specifies a command or script that the sensor resource manager will use to set the attribute values of the sensor. You should not call any of the sensor resource manager commands (**chsensor**, **lssensor**, **mksensor**, **refsensor**, or **rmsensor**) as part of this parameter.

If *sensor_command* contains any blank characters, or any special characters that can be interpreted by the shell, it must be enclosed in double quotation marks.

When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**). See Example 2 for more information.

sensor_name
Specifies the name of the sensor that is to be defined.

Security

To create sensors using this command, you need write permission for the **IBM.Sensor** resource class.

To create microsensors using this command, you need write permission for the **IBM.MicroSensor** resource class.

Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* 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.

n Based on other errors that can be returned by the RMC subsystem.

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_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

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.

Restrictions

You should not call any of the sensor resource manager commands (**chsensor**, **lssensor**, **mksensor**, **refsensor**, or **rmsensor**) as part of the *sensor_command* parameter, as this could cause a deadlock.

Implementation Specifics

This command is part of the **rsct** fileset for the AIX operating system and **rsct-3.1.0.0-0.***platform***.rpm** package for the Linux, Solaris, and Windows platforms, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Examples

1. To create a new sensor called **Sensor1** that runs the script **/usr/bin/updateSensor1**, which will update the sensor attributes every 30 seconds (once monitored), enter:

   ```
   mksensor -i 30 Sensor1 "/usr/bin/updateSensor1"
   ```

   The contents of **/usr/bin/updateSensor1** may be like:

   ```
   #!/usr/bin/perl
   my $int32 = some_fn_that_generates_i32_value;
   my $string = some_fn_that_generates_string_value;
   print "Int32=$int32 String=$string";
   exit 0;
   ```

   A sample condition could be:

   ```
   mkcondition -r IBM.Sensor -s "Name==Sensor1" -e "Int32 > 100" Sensor1Int32
   ```

   Using the response "E-mail root anytime", a start monitoring command may be:

   ```
   startcondresp Sensor1Int32 "E-mail root anytime"
   ```

2. To create a sensor called **Sensor1** with a *sensor_command* value of

   df -m /var | sed '1d' | sed 's/%//g' | /bin/awk '{ print "Int32="$4}',

enter:

```
mksensor Sensor1 "df -m /var | sed '1d' | sed 's/%//g' | /bin/awk \
'{ print \"Int32=\"\$4}'"
```

When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**). So in this example, the sensor command substring **"Int32="$4** becomes **\"Int32=\"\$4** when it is part of **mksensor** command.

3. To create a sensor called **Sensor3** that runs the **/usr/bin/checkhealth** script on the nodes that are listed in the **/u/joe/common_nodes** file, enter:

   ```
   mksensor -N /u/joe/common_nodes Sensor3 "/usr/bin/checkhealth"
   ```

   where **/u/joe/common_nodes** contains:

   ```
   # common node file
   #
   node1.myhost.com    main node
   node2.myhost.com    backup node
   ```

4. To create a microsensor called **IBM.msensorq** that uses the shared module **/usr/lib/msensors/msensorq** and requires the parameters **db=abc**, **confirm=yes**, **retry=yes**, and **mirror=no**, enter:

   ```
   mksensor -m IBM.msensorq /usr/lib/msensors/msensorq \
   "db=abc confirm=yes retry=yes mirror=no"
   ```

Location

/opt/rsct/bin/mksensor

---

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

Item	Description
-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

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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.

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

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration object class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration object class.

Related information:

auditpr command

chserver command

Setting up auditing

Defining Your Subsystem to the SRC

---

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 System Management Interface Tool (SMIT) **smit mkslave** fast path to run this command.

Flags

Item	Description
-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

Item	Description
/var/yp/*DomainName* directory	Contains the NIS maps for the NIS domain.

Related information:

rmyp command

ypinit command

yppasswdd command

ypserv command

---

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

Item	Description
-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

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges,

see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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.

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

Item	Description
/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:

auditpr command

refresh command

traceson command

Setting up auditing

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

Item	Description
-	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

Item	Description
/usr/ccs/bin/mkstr	Contains the **mkstr** command.

Related information:

cat command

xstr command

lseek command

---

mksysb Command

Purpose

Creates an installable image of the root volume group either in a file or onto a bootable tape.

Syntax

mksysb [**-a**] [**-A**] [**-b** *number*] [**-e**] [**-F** *filename*] [**-i**] [**-m**] [**-p**] [**-P**] [**-t** *argument*] [**-v**] [**-V**] [**-x** *file*] [**-X**] [**-Z**] [**-G** | **-N**] [**-M**] [**-T**] *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 if it is corrupted. If you create the backup on tape or user defined file system (UDFS) capable media, the backup is bootable and includes the installation programs that are needed to install from the backup.

Note: If the system has a multibos environment where both instances are mounted, you can restore the backup only by using the **alt_disk_mksysb** command.

You can also use a **mksysb** image to restore another system.

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 that the **mksysb** command uses is the `/bosinst.data` file. If the `/bosinst.data` file does not exist, the `/var/adm/ras/bosinst.data` file is copied to / (root). The **mksysb** command always updates the `target_disk_data` stanzas in the `bosinst.data` file 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 that are updated, you must create /save_bosinst.data_file. The **mksysb** command does not update /bosinst.data if the /save_bosinst.data_file exists.

Notes:

1. When the **mksysb** command is running, ensure that system activity is minimal.
2. The image that the **mksysb** command creates does not include data on raw devices or in user-defined paging spaces.
3. If you are using a system with a remote-mounted /usr file system, you cannot reinstall your system from a backup image.
4. The **mksysb** command might not restore all device configurations for special features, such as /dev/netbios and some device drivers that are not shipped with the product.
5. The **mksysb** command uses the **backup** command to create an archive image. The **mksysb** command also saves the extended attributes (EA) format for any Enhanced Journaled File System (JFS2) that are being backed up. It uses the /usr/bin/mkszfile shell script to save this information.
6. If you remove the /dev/ipldevice before running the **mksysb** command, the 0301-150 bosboot error occurs. This message, in most cases, can be ignored. Confirm the success of the **mksysb** command by the return code.
7. If you are creating a tape backup and have encrypted file systems, you must use the **-Z** flag. You cannot reinstall your system from a tape backup image that contains encrypted file systems.

To create a backup of the operating system to a CD, refer to the "mkcd Command" on page 667. To create a backup of the operating system to a DVD, refer to the "mkdvd Command" on page 707.

Flags

Item	Description
-a	Does not back up extended attributes or Network File System version 4 (NFS4) access control lists (ACLs).
-A	Backs up Data Management API (DMAPI) file system files.
-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 that are multiplied by the block size. The default write size for the **backup** command that writes 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.

Item	Description

-e	Excludes files that are 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 that are 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:
	1. To exclude all the contents of the directory that is called scratch, edit the exclude file to read as follows:
	`/scratch/`
	2. To exclude the contents of the directory that is called /tmp, and avoid excluding any other directories that contain /tmp in the path name, edit the exclude file to read as follows:
	`^./tmp/`
	
	This excludes all contents of the **/tmp** directory, but the **/tmp** mount point for the file system is retained. It does not remove other directories or its contents such as **/var**, **/adm**, **/sw**, and **/tmp** file system.
	
	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 the ^ (caret character) as the first character in the search string, followed by **.** (dot character), followed by the file name or directory to be excluded.
	
	If the file name or directory that is being excluded is a substring of another file name or directory, use the **^.** (caret character followed by dot character) to indicate that the search must begin at the beginning of the line and use the **$** (dollar sign character) to indicate that the search must end at the end of the line.

-F *filename*	Specifies a previously created **mksysb** image from which a backup tape is created. An attempt is made to make the backup tape bootable. Additionally, this flag must be used with a tape device.
-G	Excludes WPAR file systems from the system backup. This flag is not valid with **-N** flag.
-i	Calls the **mkszfile** command, which generates the /image.data file. The /image.data file contains details about 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. You must use the **-i** flag. Otherwise, an older /image.data file might be saved that does not contain adequate space requirements to restore the system backup. **Note:** Before you run 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 that is needed in the /tmp file, enter the following command: `bosboot -q -a -d device` If you are using UDFS capable device named /dev/usbms0, you must specify /dev/cd0 as the device name because the /dev/usbms0 device is not supported by the **bosboot** command. 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.
-M	Creates a backup file that is intended for use with the **multibos** command. The **-M** flag backs up the **/**, **/usr**, **/var**, and **/opt** file systems. Do not use the backup to reinstall a system. You must install the bos.alt_disk_install.boot_images fileset at the same level as the system.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the system backup. **Note:** To be included in the backup, all file systems that belong to a WPAR in the defined state must be in the rootvg volume group.
-p	Disables software packing of the files as they are backed up. Some tape drives use their own packing or compression algorithms.

Item	Description
-P	Excludes files that are listed line by line in the /etc/exclude_packing.rootvg, /etc/exclude_packing.vgname, or /etc/exclude_packing.WPARname file from being packed. For example, to exclude the /etc/filesystems and /usr/bin/zcat file from being packed during the **mksysb** backup, edit the /etc/exclude_packing.*type* to add on consecutive lines /etc/filesystems and /usr/bin/zcat. In this case, the file /etc/exclude_packing.*type* must look like: /etc/filesystems /usr/bin/zcat The **-P** and **-p** flags are mutually exclusive.
-t *argument*	Specifies the path to the directory or file system that is used to create a boot image from the **mksysb** file that is specified by the **-F** flag. If the **-t** flag is not used with the **-F** flag, the boot image is created in the **/tmp** file by default. Approximately 100 MB of free space is required. After the boot image is created, this space is freed.
-T	Creates backup by using snapshots. This command applies only to JFS2 file systems. When you specify the **-T** flag to use snapshots for creating a volume group backup, external JFS2 snapshots are created. Snapshots allow for a point-in-time image of a JFS2 file system and thus, do not require a system to be put into a temporarily inactive state. The size of the snapshot is 2% - 15% of the size of the file system. The snapshot logical volumes are removed when backup is complete. However, snapshots are not removed if a file system already has other snapshots. Additionally, if a file system has internal snapshots, external snapshots cannot be created and thus, snapshots are not used for creating the backup of the file system. The use of the **-T** flag does not affect any JFS file systems that are present in the volume group that is being backed up. These file systems are backed up in the same manner as done previously. When you specify the **-T** flag, you must also specify the **-i** flag. If you do not specify the **-i** flag, an older /image.data file might not have adequate space requirements. Therefore, a failure might occur when you save data to the snapshot.
-v	Verbose mode. Lists files as they are backed up.
-V	Verifies a tape backup. This flag causes the **mksysb** command to verify the file header of each file on the backup tape and report any read errors as they occur.
-x *file*	Excludes the file systems that are listed in the file from the system backup. File system mount points must be listed one per line.
-X	Specifies to automatically expand the **/tmp** file system if necessary. The **/tmp** file system might need to be extended to make room for the boot image when creating a bootable backup to tape.
-Z	Specifies that the Encrypted File System (EFS) information for all the files, directories, and file systems is not backed up.The **-Z** flag is required if you have encrypted file systems and are creating a backup on a tape.

Attention: Use the **-x** flag with caution when you exclude the file systems from the backup of the operating system. The resulting backup might be unusable for system restoration.

Parameters

Item	Description
Device | *File*	Specifies the name of the device or file.

Exit Status

This command returns the following exit values:

Item	Description
0	The command completed successfully.
1	An error in the **mksysb** command use occurred.
2	An error in the **savevg** command use occurred. The **savevg** is a link to **mksysb**.
3	An error occurred before any file systems were mounted.
4	Exit because of trap.
5	Exit because there was no space.
6	Exit because a volume group name was not valid.

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, enter the following command:

   ```
   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, enter the following command:

   ```
   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 enter the following command:

   ```
   mksysb -i -e /dev/rmt1
   ```

   This command backs up 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, enter the following command:

   ```
   mksysb -i /mksysb_images/node1
   ```

   Note: This file is not bootable and can be installed only by using Network Installation Management (NIM).

5. After running the **mkszfile** command independently, to generate a system backup on the tape device /dev/rmt0, and then to verify the readability of file headers, enter the following command:

   ```
   mksysb  /dev/rmt0 -V
   ```

6. To generate a system backup file named /mksysb_images/mksysb1 to be used with the **multibos** command, and to create an /image.data file for that image, enter the following command:

   ```
   mksysb -iM /mksysb_images/mksysb1
   ```

7. To generate a system backup and create an /image.data file (generated by the **mkszfile** command) to a UDFS capable device named /dev/usbms0, enter the following command:

   ```
   mksysb -i /dev/usbms0
   ```

   Note: For information backing up a volume group, see the **listvgbackup** command. To restore individual files from a volume group backup, see the **restorevgfiles** command.

Files

Item	Description
/usr/bin/mksysb	Contains the **mksysb** command.

Related reference:

"mkszfile Command"

Related information:

backup command

/image.data file

Creating system backups

Cloning a system backup

---

mkszfile Command

Purpose

Saves the system state for reinstallation on the current system or another system.

Syntax

mkszfile [**-X**] [**-d** directory] [**-m**] [**-G**] [**-N**] [**-x** file]

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:

1. 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 one of the following commands: **bosboot -qad** *rmt* or **bosboot -qad** *ipldevice*.
2. If you remove the `/dev/ipldevice` prior to executing the **mkszfile** command, the 0301-150 bosboot error occurs. This message, in most cases, can be ignored. Confirm the success of the **mkszfile** command by the return code.

Flags

Item	Description
-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.
-N	Includes file systems that belong to a workload partition (WPAR) in the defined state in the **/image.data** file. **Note:** To be included in the **/image.data** file, all file systems that belong to a WPAR in the defined state need to be in the rootvg volume group.
-X	Expands **/tmp** if needed.

Item	Description
-d	Write the **image.data** file to the specified directory instead of /.
-G	Excludes the WPAR file systems from the **/image.data** file. This flag is not valid with **-N** flag.
-x *file*	Excludes the file systems that are listed in the file from the **image.data** file. File system mount points must be listed one per line.

Note: Use care when excluding file systems as a resulting backup can be unusable for system restoration.

Files

Item	Description
/usr/bin/mkszfile	Contains the **mkszfile** command.

Related reference:
"mksysb Command" on page 829
Related information:
/image.data command

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** *DefaultGateway*] [**-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.

Note: The **mktcpip** command currently supports IPv4 only.
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 System Management Interface Tool (SMIT) **smit mktcpip** fast path to run this command.

Flags

Item	Description
-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 *DefaultGateway*	Adds the default gateway address to the routing table. Specify the address in dotted decimal notation. The following is the standard format for setting the default gateway address: `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
```

Note: Use the **mktcpip** command only to minimally configure TCP/IP for the first time. For further configuration changes, use the **smitty configtcp** fastpath.

Files

Item	Description
/usr/bin/mktcpip	Contains the **mktcpip** command.
/etc/resolv.conf	Contains the default system configuration database.
/etc/hosts	Contains the host name and IP address entries.

Related information:

hostname command

hostent command

resolv.conf file

TCP/IP addressing

mkts Command

Purpose

Makes a thin server.

Syntax

mkts -i *ipaddress* **-m** *subnetmask* **-g** *gateway* [**-s** *speed*] [**-d** *duplex*] **-c** *cosi* [**-p** *size*] [**-H** | **-h**] [**-t**] [**-l**] [**-v**] [**-D**] *thinserver*

Description

The **mkts** command creates a thin server so that it can use the common image created with the **mkcosi** command. When a thin server is created, several directories are also created for the thin server to mount and use, including **/root**, **/dump**, **/home**, **/tmp**, **/shared_home**, and **/paging**. If you specify the **-l** flag when creating a thin server, the resulting thin server is a diskless client. That is, all resources are created on the server that calls the **mkts** command, except for the **/root** directory, which is created on the server storing the common image. However, if you do not specify the **-l** flag, the thin server is a dataless client. In this case, only the **/root** directory is created on the server storing the common image; all other directories are created locally on the thin server. If necessary, the 512 MB default size used for the paging can be changed by specifying a size value with the **-p** flag.

Flags

Item	Description
-c *cosi*	Specifies the common image for the thin server to obtain its operating system, which is required for the thin server to start up and run.
-d *duplex*	Specifies the duplex setting (optional). Use this setting to configure the client's network interface. This value can be full or half.
-D	Creates an iSCSI dump device in the Common Operating System Image (COSI) of the thin server. The name of the dump device is dump_*cosi_name*. In order to allow the **sysdumpdev** command to correctly set this new iSCSI dump device on the thin server, the thin server must boot in iSCSI mode from the COSI of the thin server.
-g *gateway*	Specifies the thin server gateway.
-h	Defines or uses home resource. The home resource is a network installation management (NIM) home resource. It is a directory that is created on a NIM master or any NIM resource server. The directory is exported to the thin server to be mounted and used. It is basically the thin server's **/home** directory.

Item	Description
-H	Defines or uses shared_home resource. The shared_home resource is a network installation management (NIM) resource. It is a directory that is shared among all thin servers. The directory is exported and mounted on the clients from the NIM master.
-i *ipaddress*	Specifies a thin server IP address or host name.
-l	Specifies whether local resources are used when configuring the thin server. If you specify this flag, all resources are created remotely from the thin server. If you do not specify this flag, only the **/root** resource is created remotely from the thin server, and all other resources are created locally on the thin server.
-m *subnetmask*	Specifies the thin server subnet mask.
-p *size*	Specifies the size (in megabytes) of the paging space for the thin server. The minimum size is 64 MB of paging space. The default size is 512 MB of the paging space. If you specify the size less than 64 MB, 512 MB is used.
-s *speed*	Specifies speed setting (optional). This is the communication speed to use when configuring the client's network interface. This value can be 10, 100, or 1000.
-t	Defines or uses the TMP resource.
-v	Enables verbose debug output when the **mkts** command runs.

Exit Status

Item	Description
0	The command completed successfully.
>0	An error occurred.

Security

Access Control: You must have root authority to run the **mkts** command.

Examples

1. To define a thin server called `lobo` and have it use a common image called `cosi1` for its operating system with an IP address of 9.3.6.234, a subnet mask of 255.255.255.0, and a gateway of 9.3.6.1, enter:

   ```
   mkts -i 9.3.6.234 -m 255.255.255.0 -g 9.3.6.1 -c cosi1 lobo
   ```

Location

/usr/sbin/mkts

Files

Item	Description
/etc/niminfo	Contains variables used by NIM.

Related information:

dbts command

nim_clients_setup command

nimconfig command

swts command

mktun Command

Purpose

Activates tunnel(s).

Syntax

mktun [**-v 4**|**6**] [**-t** *tid_list*] [**-i**] [**-I**]

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

Item	Description
-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 status in the tunnel database "active" will be activated.
-I	If the **-I** 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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Related reference:

"imptun Command" on page 35

Related information:

chtun command

exptun command

gentun 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 string (whose length is administrator-configurable using the **chdev** command) . 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, you can use the **-R** flag to specify the I&A load module. If you create users without the **-R** flag, you create the users locally. Load modules are defined in the **/usr/lib/security/methods.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 System Management Interface Tool (SMIT) **smit mkuser** fast path to run this command.

The **mkuser** command always checks the target user registry to make sure the ID for the new account is unique to the target registry. You can also configure the **mkuser** command to check all user registries of the system using the **dist_uniqid** system attribute. The **dist_uniqid** system attribute is an attribute of the **usw** stanza of the **/etc/security/login.cfg** file, and can be managed using the **chsec** command.

The **dist_uniqid** system attribute has the following values:

- **never** - Does not check for ID collision against the non-target registries. This is the default setting.
- **always** - Checks for ID collision against all other registries. If collision is detected between the target registry and any other registry account creation or modification fails.
- **uniqbyname** - Checks for ID collision against all other registries. Collision between registries is allowed only if the account to be created has the same name as the existing account.

Note: ID collision detection in the target registry is always enforced regardless of the **dist_uniqid** system attribute.

The **uniqbyname** system attribute setting works well against two registries. With more than two registries, and with ID collision already existing between two registries, the behavior of the **mkuser** command is unspecified when creating a new account in a third registry using colliding ID values. The new account creation might succeed or fail depending the order in which the registries are checked.

The check for ID collision only enforces ID uniqueness between the local registry and remote registries or between remote registries. There is no guarantee of ID uniqueness between the newly created account on the remote registry and existing local users on other systems that make use of the same remote registry. The **mkuser** command bypasses a remote registry if the remote registry is not reachable at the time the command is run.

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:

Item	Description
:	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

Item	Description
-a	Specifies that the user is an administrator. Only the root user can use this flag or alter the attributes of an administrative user.
username	Specifies that the user is a new user.
-R *load_module*	Specifies the loadable I&A module used to create the user.

Parameters

Item	Description
Attribute=*Value*	Initializes a user attribute. Refer to the **chuser** command for the valid attributes and values.
Name	Specifies a unique string. The length of this string is set by an administrator by using the **chdev** command.

Exit Status

This command returns the following exit values:

Item	Description
0	The command runs successfully and all requested changes are made.
>0	An error occurred. The printed error message lists further details about the type of failure.

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.

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.
To get the full functionality of the command, besides the **accessauths**, the role should also have the following authorizations:

- **aix.security.user.audit**
- **aix.security.role.assign**
- **aix.security.group.change**
- **aix.security.user.change**

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
   ```

Error Codes

Item	Description
0	The command is successful.
EINVAL	The username argument is not valid (containing characters not valid).
EACCES	The invoker does not have write access to the database files.
EPERM	The user identification and authentication fails if the **-a** flag is specified and the invoker is not root.
EEXIST	The user already exists.
ENAMETOOLONG	The user name is too long.
other **errno**	There are other system errors.

Files

Item	Description
/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 reference:

"lsgroup Command" on page 412

"mkuser.sys Command"

Related information:

pwdadm command

setgroups command

PowerHA SystemMirror Administration Guide

mkuser.sys Command

Purpose

Customizes a new user account.

Syntax

mkuser.sys *Directory User Group Shell*

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 **tsm**, **login**, and **getty** commands and the **pam_mkuserhome** module call the **mkuser.sys** command at your login time if you do not have a home directory already.

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.

Note: The shipped **mkuser.sys** file must not be customized directly. If a customized version is required, a new file **/etc/security/mkuser.sys.custom** must be created. The **mkuser.sys** program detects this new program and if it is present on the system, it runs it instead of the original **mkuser.sys**. The shipped **mkuser.sys** file is now a non-volatile file and must not be modified. The installation-specific program must 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.

Files Accessed:

Mode	File
r	/etc/passwd
r	/etc/security/user

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/lib/security/mkuser.sys	Contains the **mkuser.sys** command.

Note: You cannot use the **/etc/security/mkuser.sys** file to edit with the **chuser** , and **rmuser** commands. To assign default attributes such as *primary group*, *home directory* , and *login shell* to a user, use the **/etc/security/mkuser.default** file.

Related reference:

"mkuser Command" on page 839

"login Command" on page 253

Related information:

tsm command

getty command

Trusted AIX®

mkusil Command

Purpose

Creates or attaches a new user-specified installation location (USIL) instance.

Syntax

mkusil -R *RelocatePath* **-c** *Comments* [**-XFa**]

Description

The **mkusil** command creates or attaches a new USIL instance.

A user-specified installation location (USIL) is a tracked, relocated installation path that is created by the administrator. The location is tracked by the system and can be used as an alternate installation path for packages that use relocation of file sets or software. Multiple instances or versions of the same software package can be installed on a single system by delegating each installation to a separate USIL. An existing USIL instance can be attached or detached from any given system.

Each USIL instance maintains its own set of Software Vital Product Data (SWVPD) in three **installp** parts:

- *InstallRoot***/etc/objrepos**
- *InstallRoot***/usr/lib/objrepos**
- *InstallRoot***/usr/share/lib/objrepos**

Tip: Current SWVPD object classes include product, lpp, inventory, history, fix, vendor, and lag. Each USIL instance mirrors the default SWVPD structure within the relocated path.

Flags

Item	Description
-a	Attaches an existing installation as a USIL instance.
-c *Comments*	Specifies the comments to include in the USIL definition.
-F	Overwrites the existing USIL SWVPD in the target path without prompting you. It is appropriate to use this flag for USIL SWVPDs that are detached or removed by the **rmusil** command.
-R *RelocatePath*	Specifies the path to a new USIL location, which must be a valid directory. The directory cannot belong to a file system with the CIO mount option.
-X	Expands the space needed automatically.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Files

Item	Description
/usr/sbin/mkusil	Contains the *mkusil* command.

Related reference:

"lsusil Command" on page 553

Related information:

chusil command

rmusil command

---

mkvg Command

Purpose

Creates a volume group.

Syntax

mkvg [**-B**] [**-t** *factor*] [**-S** [**-v** *logicalvolumes*] [**-P** *partitions*]] [**-C**] [**-G**] [**-f**] [**-i**] [**-I**][**-c**] [**-X** *none | SSD*] [**-L** *ltgsize*] [**-n**] [**-s** *size*] [**-V***majornumber*] [**-y***volumegroup*] [**-M** **y**|**s**] [**-p** *mirrorpool*] [**-O** *y | n*] [**-N** *o*|*n*] [**-r** *y*|*n*] [**-e** *y*|*n*] *physicalvolume ...*

Description

The **mkvg** command creates a new volume group, by using the physical volumes that are represented by the *physicalvolume* parameter. After creating the volume group, the **mkvg** command automatically varies on the new volume group by 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 is not 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 by specifying either the **-B** or **-S** flag.

The **mkvg** command attempts to determine a proper partition size (**-s**) and factor (**-t**) if none is specified on the command line.

Note:

1. The physical volume is checked to verify that it is not already in another volume group. If the **mkvg** command determines that the physical volume belongs to a volume group that is varied on, it exits without creating the volume group. If the **mkvg** command determines the physical volume belongs to a volume group that is not varied on, the force option (**-f**) must be used to create the volume group. When using the force option, the previous contents of the physical volume are lost, so the user must use caution when using the force option.
2. To use this command, you must either have root user authority or be a member of the **system** group.
3. When creating the default volume group type (with a maximum of 32 PVs) or the big volume group type (with a maximum of 128 PVs), there is a limitation of 1016 physical partitions per PV. When specifying the physical partition size (**-s**), make sure that the value is set large enough so that 1016 physical partitions per PV limit is not violated. For example, a partition size of at least 16 MB would be needed to create a volume group with a 10-GB disk. Using a factor size (**-t**) of 2, a smaller partition size of 8 MB can be used. If a factor value is specified, the maximum number of PVs that can be included in the volume group is MaxPVs/factor.
4. 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 does 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 fails 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 manager use **chpv -C** *HDiskName*.
6. Only Enhanced Concurrent Capable volume groups are created when the **-c** or **-C** flags are specified.
7. You must not mix 4 KB block physical volumes (PV) with PV blocks of other sizes. The block size of all PVs in the volume group must be the same. You can not import the volume group that is created with 4 KB block PVs on a version of AIX that does not support 4 KB block PVs.
8. Bad block relocation policy of a volume group is not supported on a volume group that is created with 4 KB block PVs.

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 System Management Interface Tool (SMIT) **smit mkvg** fast path to run this command.

Flags

Item	Description
-B	Creates a **big** type of volume group. This type 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 logical volume, adding a physical volume, and so on) might take considerably longer to run.
-c	Same as **-C** flag. Only Enhanced Concurrent Capable volume groups are created.

Item	Description
-C	Creates an Enhanced Concurrent Capable volume group. Only use the **-C** flag with the PowerHA® SystemMirror® enhanced scalability (ES). It is not usable on volume groups and systems that do not use the PowerHA SystemMirror ES product. Use this flag to create an Enhanced Concurrent Capable volume group. **Note:** 1. Enhanced Concurrent volume groups use Group Services. Group Services ships with PowerHA SystemMirror ES and must be configured before 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. 3. Enhanced Concurrent Capable volume groups have multinode varyon protection enabled. See the **-N** flag for details about multinode varyon protection.
-e *y* | *n*	Enables the `Critical PVs` option of the volume group. This flag is available in IBM AIX 7.2 with Technology Level 1, or later. **y** Enables the `Critical PVs` option of the volume group. If write request failures occur in the mirrored logical volume, the PV is marked as missing and it stops sending I/O requests to the failed mirrored logical volume. If the `Critical PVs` option is enabled in a volume group, you can import only the volume group into IBM AIX 7.2 with Technology Level 1, or later. **n** The `Critical PVs` option is not used. This is the default value.
-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	Same as **-B** flag.
-i	Reads the *PhysicalVolume* parameter from standard input.
-I	Creates a volume group that can be imported to AIX Version 6.1. The *LTGSize* behaves as if the volume group had been created before AIX Version 6.1. If the logical volumes are later created with a strip size that is larger than the supported strip size on AIX Version 6.1 or AIX Version 6.1 (a strip size that is multiplied by the number of disks in an array equals the stripe size), then attempting to import the volume group back to AIX Version 6.1 or AIX Version 6.1 is not supported.
-L *ltgsize*	For volume groups created on AIX Version 6.1 without the **-I** flag, the **-L** flag is ignored. When the volume group is varied on the logical track group size is set to the common maximum transfer size of the disks. For volume groups created on AIX Version 6.1 with the **-I** flag or for volume groups that are created before AIX Version 6.1, the logical track group size is set to the *ltgsize*, which must be `128`, `256`, `512`, or `1024`. In addition, it must be less than or equal to the maximum transfer size of all disks in the volume group. The default *ltgsize* is 128-KB.
-M y | **s**	Enables mirror pool strictness for the volume group. **y** Mirror pools must be used on each logical volume in the volume group. **s** Super-strict mirror pools are enforced on this volume group.
-N *o* | *n*	**o** Creates a volume group that is allowed to varyon in non-concurrent mode in more than one node at the same time. This is the default value. **n** Creates a volume group that is not allowed to varyon in non-concurrent mode in more than one node at the same time. This volume group can no longer be imported on a version of the AIX operating system that does not support the **-N** flag.
-n	Specifies that the volume group is not automatically available during a system restart. The default value activates the volume group automatically.
-O *y* / *n*	Enables the infinite retry option of the logical volume. **n** The infinite retry option of the logical volume is not enabled. The failing I/O of the logical volume is not retried. This is the default value. **y** The infinite retry option of the logical volume is enabled. The failed I/O request is retried until it is successful. **Note:** The infinite retry option is not supported in the Geographic Logical Volume Manager (GLVM) environment.

Item	Description
-p *mirrorpool*	Assigns each of the physical volumes that are being added to the specified mirror pool. After mirror pools are enabled in a volume group, the volume group can no longer be imported into a version of AIX that does not support mirror pools.
-P *partitions*	Total number of partitions in the volume group, where the *Partitions* variable is represented in units of 1024 partitions. Valid values are 32, 64, 128, 256, 512 768, 1024 and 2048. The default is 32 k (32768 partitions). The **chvg** command can be used to increase the number of partitions up to the maximum of 2048 k (2097152 partitions). This option is only valid with the **-S** option.
-r *y*|*n*	Enables the `Critical VG` option of the volume group. The **-r** flag can have the following values: **y** The `Critical VG` option of the volume group is enabled. If the volume group is created with the `Critical VG` option turned on, any I/O request failure starts writing the logical volume manager (LVM) metadata to check the state of the disk before returning the I/O failure. If the rootvg volume group is set to the `Critical VG` option and if the volume group looses access to quorum set of disks (or all disks if quorum is disabled), instead of forcing the volume group to an offline state, the node crashes and a message is displayed on the console. **n** The `Critical VG` option of the volume group is not enabled. It is the default value.
-S	Creates a **scalable** type of volume group. By default, this volume group can accommodate up to 1024 physical volumes, 256 logical volumes, and 32768 physical partitions. To increase the number of logical volumes, use the **-v** option. To increase the number of physical partitions, use the **-P** option. **Note:** Increasing **maxlvs** and **maxpps** beyond the default values for a scalable volume group can significantly increase the size of the VGDA proportionately. The **maxlvs** and **maxpps** values must be increased only as needed because they cannot be decreased. Meanwhile, as the VGDA space increases all VGDA update operations (creating a logical volume, changing a logical volume, adding a physical volume, and so on) can take longer and longer to run.
-s *Size*	Sets the number of megabytes in each physical partition, where the *Size* variable is expressed in units of megabytes from 1 (1 MB) through 131072 (128 GB). The *Size* variable must be equal to a power of 2 (example 1, 2, 4, 8). The default value for 32 and 128 PV volume groups is the lowest value to remain within the limitation of 1016 physical partitions per PV. The default value for scalable volume groups is the lowest value to accommodate 2040 physical partitions per PV.
-t *factor*	Changes the limit of the number of physical partitions per physical volume, which is specified by *factor*. The *factor* must be 1 - 16 for 32 PV volume groups and 1 and 64 for 128 PV volume groups. The maximum number of physical partitions per physical volume for this volume group changes to *factor* x 1016. The default is the lowest value to remain within the physical partition limit of *factor* x 1016. The maximum number of PVs that can be included in the volume group is MaxPVs/factor. The **-t** option is ignored with the **-S** option.
-V *majornumber*	Specifies the major number of the volume group that is created.
-v	Number of logical volumes that can be created. Valid values are 256, 512, 1024, 2048 and 4096. The default is 256. The **chvg** command can be used to increase the number of logical volumes up to the maximum of 4096. This option is only valid with the **-S** option. The last logical volume is reserved for metadata.
-Xnone|**SSD**	Enables PV type restriction for the volume group. This option allows a volume group to be created with a specific restriction based on the PV type. "none" is the default value. "SSD" requires that all PVs in the volume group must be SSD media type PVs. When the PV restriction is turned on, the **mkvg** command verifies that all PVs meet this condition. Once a PV restriction is turned on, the volume group can no longer be imported on a version of AIX that does not support PV type restrictions. **none** As an option, there is no PV restriction. volume group can be formed on any disk type. This is the default value. **SSD** As an option, volume group is restricted to SSD PV types. Disks that are listed in physical volume argument must be of type SSD.

Item	Description
-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 that is created is sent to standard output. The volume group name can only contain only 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.

Security

Note:

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To create a volume group that contains 3 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 2-MB and maximum number of physical partitions per physical volume of 2032. The configuration mentioned in the example means that the size of hdisk1 can not be larger than 4064-MB (2032*2)
2. To create a volume group that can accommodate a maximum of 1024 physical volumes and 2048 logical volumes, type:

   ```
   mkvg -S -v 2048 hdisk6
   ```

Files

Item	Description
/usr/sbin	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 reference:

"lsvg Command" on page 554

Related information:

chvg command

varyonvg command

Logical volume storage

PowerHA SystemMirror Administration Guide

mkvgdata Command

Purpose

Creates a file containing information about a volume group for use by the **savevg** and **restvg** commands.

Syntax

mkvgdata [**-X**] [**-m**] [**-x** *file*] *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:

Item	Description
/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

Item	Description
-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.
-x *file*	Excludes the file systems that are listed in the file from the output file. File system mount points must be listed one per line.

Note: Use care when excluding file systems because a resulting backup can be unusable for system restoration.

Files

Item	Description
/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 reference:
"mkszfile Command" on page 833
Related information:
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 System Management Interface Tool (SMIT) **smit mkvirprt** fast path to run this command.

Flags

Item	Description
-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.
-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

Item	Description
/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 reference:

"lp Command" on page 266

"mkque Command" on page 780

Related information:

chvirprt command

qprt command

/etc/qconfig file

Installing support for additional printers

mkwpar Command

Purpose

Creates a system workload partition (WPAR), or a WPAR specification file.

Syntax

/usr/sbin/mkwpar [**-a**] [**-A**] [**-b** *devexportsfile*] [**-c**] [**-C**] [**-E** *directory*] [**-d** *directory*] [**-B** *wparbackupdevice*] [**-D** *attribute=value* ...] ... [**-F**] [**-g** *vg*] [**-h** *hostname*] [**-H** *architecture*] [**-i**] [**-I** *attribute=value* ...] [**-k**]... [**-l**] [**-L** *attribute=value*...] [**-M** *attribute=value* ...] ... [**-N** *attribute=value* ...] ... [**-P**] [**-r**] [**-R** *attribute=value* ...] [**-S** *attribute*[+|-]=*value* ...] [**-t**] [**-T** attribute=value ...] [**-s**] [**-u** *userscript*] [**-X** *attribute=value* ...] [**-U** [*uuid*]] { **-n** *wparname* [**-p** [*name*]] [**-e** *existingwparname* **-W** | **-f** *infile*] [**-o** *outfile* [**-w**]] | **-p** [*name*] [**-n** *wparname*] [**-e** *existingwparname* **-W** | **-f** *infile*] [**-o** *outfile* [**-w**]] | **-f** *infile* [**-n** *wparname*] [**-p** [*name*]] [**-o** *outfile* [**-w**]] | **-w -o** *outfile* [**-n** *wparname*] [**-p** [*name*]] [**-e** *existingwparname* **-W** | **-f** *infile*] }

Restriction:

- White space must be included between a flag and its argument for *attribute=value* type flags. The **mkwpar** command is not supported on the TCB systems. Regardless of locale, only ASCII characters are allowed as arguments to **mkwpar**, **chwpar**, or **wparexec**.
- You must not run the **mkwpar** command during the AIX Live Update operation.

In addition to the previous command restrictions, more restrictions follow for the WPAR name:

- Must not be more than 25 bytes.
- Must not contain white space or any of the following symbols:
  `= : / ! ; ` ' " < > ~ & ( ) * + [ ] , . ^ 0 { } | \`
- Must not start with hyphen (-) or 0.

Description

The **mkwpar** command builds the infrastructure to prepare a system workload partition for use. This command includes the following tasks:

- Creating the configuration data of the workload partition in the workload partition database
- Creating and populating file systems of the workload partition

- Creating an SRC subsystem for the init process of the workload partition
- Defining the resource control profile of the workload partition through Workload Manager

The following options are also available:
- Writing a specification file to simplify creation of other, similar workload partitions
- Starting the workload partitions
- Specifying whether the workload partitions must be automatically started on system start or when **/etc/rc.wpars** is started
- Specifying WPAR specific routing, by using the **-i** and **-I** flags

The **mkwpar** command supports advanced logical volume and file system options by specifying the **image.data** file as an argument to the **mkwpar -L** flag.

The **mkwpar** command supports creating a rootvg WPAR, in which the root file systems are located solely in WPAR storage devices.

Flags

Item	Description
-a	Automatically resolves conflicting static settings if required. Resolvable settings are base directory, host name, and network configuration.
-A	Specifies that the workload partition must be started each time **/etc/rc.wpars** is run, which is added to the global **/etc/inittab** to run each time that the system starts. The default is not to start the workload partition automatically. **Tip:** The workload partition is started immediately upon completion of the **mkwpar** command. To start the workload partition immediately, use the **-s** flag.
-b *devexportsfile*	Specifies an alternative file to use as the master device exports file. This file must match the format of a Device Exports File. If you do not specify a file name, **/etc/wpars/devexports** is used.
-B *wparbackupdevice*	Specifies a device that contains a workload partition backup image. This image is used to populate the workload partition file systems. The *wparBackupDevice* parameter is a workload partition image that is created with the **savewpar**, **mkcd**, or **mkdvd** command. The **-B** flag is used by the **restwpar** command as part of the process of creating a workload partition from a backup image. **Note:** The **-B** flag is mutually exclusive with the **-p** flag.
-c	Configures the workload partition to be checkpointable. This option is valid only when more checkpoints or restart software are installed and configured. When you specify this flag, any file systems that are associated with only this flag (for example, through the **-M** flag) must be remote (for example, **vfs=nfs**).
-C	Creates a versioned workload partition. This option is valid only when more versioned workload partition software is installed.
-d *directory*	Specifies a base directory for the workload partition. If you do not specify a directory name, **/wpars/<wparname>** is used.

Item	Description
-D [**devname**=*device name* | **devid**=*device identifier*] [**rootvg**=*yes* | *no*] [**devtype**=[*clone* | *pseudo* | *disk* | *adapter* | *cdrom* | *tape*]]	Configures exporting or virtualization of a global device into the workload partition every time the system starts. You can specify more than one **-D** flag to allocate multiple devices. Separate the attribute=value by blank spaces. You can specify the following attributes for the **-D** flag: **devname=device name** Specifies the device name to allocate to the workload partition. For pseudo and clone type devices, this command is the full path to the device (that is, /dev/pty10). For storage type devices, it is the logical device short name. **devid=device identifier** Specifies the unique device identifier of a disk type device to allocate to the workload partition. This attribute applies only to disk, cdrom, or tape type devices. **devtype=[clone | pseudo | disk | adapter | cdrom | tape]** Specifies the device type of the device to allocate to the workload partition. **rootvg= [yes | no]** Used to indicate whether the specified disk device is to be used as a rootvg workload partition device. If the **rootvg** attribute is not specified, the command takes the default of number.
-e *existingwparname*	Uses an existing workload partition as the source for specification data. This flag is mutually exclusive with the **-f** flag. Any values that you specify using other **mkwpar** flags override those from the existing workload partition.
-E	Specifies a directory which contains additional filesets to install when a versioned workload partition is created. If you do not specify a directory name, `/usr/sys/inst.images` is used. This option is used only during creation of a versioned WPAR.
-f *infile*	Indicates a specification file from which default values are read. This flag is mutually exclusive with the **-e** flag. Any values that you specify by using other **mkwpar** flags, override those flags from the loaded specification file.
-F	Forces the command to continue rather than fail for most error conditions.
-g *vg*	Indicates the default volume group. If you do not specify a value, rootvg is used. This volume group is used for each localfs file system whose volume group is not specified by using the *vg* parameter of the **-M** flag. The volume group for file systems that you specified in the **image.data** file, supersedes the volume group that is specified with the **-g** flag.
-h *hostname*	Specifies a host name for the workload partition. you do not specify a value, the **mkwpar** command uses the workload partition name for the host name.
-H *architecture*	Creates an architecture compatible workload partition. The valid architecture values are *{pwr4, ppc970, pwr5, pwr6, pwr7, and pwr8}*. The architecture value must be lower than, or equal to, the system hardware level. The applications in the workload partition are presented with the lowest common denominator of the specified architecture. If the workload partition is checkpointable, the workload partition must be able to migrate between systems with hardware levels greater than, or equal to, the workload partition architecture. **Note:** Values *pwr5* and *ppc970* are not compatible with each other. You cannot create a *ppc970* compatible WPAR on a POWER5 processor-based system even though the *ppc970* processor preceded the POWER 5 processor.

Item	Description
-i	Enables WPAR specific routing for the workload partition. A default route is not created automatically. The **-I** flag is used to specify routes, including the default route.By default, outgoing network traffic from a workload partition is routed as if it is being sent from the global environment: • Traffic between addresses that are hosted on the same global system is sent through the loopback interface. • Routing table entries that are configured in the global system, including the default route, are used to transmit workload partition traffic. If you enable WPAR specific routing by specifying the **-i** flag, the workload partition creates and uses its own routing table for outgoing traffic. Routing entries are created automatically for each of the network addresses of the workload partition to accommodate broadcast, loopback, and subnet routes. For more information about the network attributes, see the **-N** flag. You can create explicit additions to the routing table of the workload partition using the **-I** flag.
-I *attribute=value ...*	Adds routing table entries to those tables that are automatically created when WPAR specific routing is in effect. You can specify more than one **-I** flag to configure multiple routes. Using the **-I** flag automatically enables WPAR specific routing as described under the **-i** flag. You can specify the following attributes with the **-I** flag: **rtdest**=*destination* — (Required) Identifies the host or network to which you are directing the route. You can specify the value by using either symbolic name or numeric address. You can use the keyword **default** to specify a default route. For more information about the route **rtdest** attribute, see the *Destination* parameter of the **route** command. **rtgateway**=*gateway* — (Required) Identifies the gateway to which packets are addressed. You can specify the value by using either symbolic name or numeric address. **rtnetmask**=*A.B.C.D* — Specifies the network mask to the destination address. **rtprefixlen**=*n* — Specifies the length of a destination prefix, which is the number of bits in the netmask. The value must be a positive integer. **rttype**={*net* | *host*} — Forces the **rtdest** attribute to be interpreted as the specified type. **rtinterface**=*if* — Specifies the interface, for example, `en0`, to associate with the route so that packets are sent by using the interface when the route is chosen. **rtfamily** ={*inet* | *inet6*} — Specifies the address family.
-k	Specifies the path to a user provided post installation customization script. The script is run in the global environment after the WPAR is created while WPAR file systems are mounted. The post customization script is called with the WPAR name as the first argument, and the WPAR base directory as the second argument. If the script exits with a nonzero return code, a warning is printed, but the **mkwpar** command has not failed.
-l	Creates private and writable versions of the **/usr** and **/opt** file systems.

Item	Description
-L [**image_data**= *imagedatafile*] [**shrink**= {**yes** | **no**}] [**ignore_maps**= {**yes** | **no**}]	**image_data** Specifies the path to the **image.data** file to be used for logical volume and file system options. The format of the **image.data** file is described in *Files Reference* and the **/usr/lpp/bosinst/image.template** file. File system specifications in the **image.data** file supersede file system specifications in the **Specifications** File. The **-c** flag and **-L image_data=** flags are mutually exclusive. **shrink** Specifies that the **LV_MIN_LPS** attribute, rather than the **LPs** attribute, must be used to determine the number of logical partitions for the logical volume. The **LV_MIN_LPS** attribute is from the **lv_data** stanzas from the file that the **image_data** attribute specifies. This attribute can minimize the amount of disk space that is required for a workload partition file system. This attribute has no effect if the **image_data** attribute is not specified. **ignore_maps** Specifies that the **MAPFILES** attribute must not be used to provide a disk mapping for the logical volumes that are associated with a workload partition. The **MAPFILES** attribute is from the **lv_data** stanzas from the file that the **image_data** attribute specifies. This attribute has no effect if the **image_data** attribute is not specified. **ignore_lvs** Specifies that the information from the **lv_data** stanzas is not used when the **image.data** file is being processed. The logical volumes are created with the default characteristics when the file systems are created. This attribute has no effect if the **image.dita** attribute is not specified. If this attribute is specified, the **ignore_maps** attribute is ignored.

Item	Description
-M directory=*dir* [**vfs**=*type*] [**size**=*sizespec*] [**vg**=*volumegroup*] [**logname**=*loglv*] [**dev**=*devicepath*] [**host**=*remotehost*] [=] [**mountopts**=*mountopts*]	Specifies mount configuration attributes. Attributes must be separated by a blank space. You can specify more than one **-M** flag. By default, the **/usr** and **/opt** file systems of the workload partition are mounted over the global **/usr** and **/opt** file systems in read-only mode. The **/proc** file system of the workload partition is mounted over the global **/proc** file system in read/write mode. New logical volumes that are created in **rootvg** for **/, /var, /tmp,** and **/home**. The default settings for a specified file system can be overridden by using the **-M** flag with the **directory** attribute set to the file system name. You can specify more file systems with additional **-M** flags. The **directory** attribute denotes the directory within the workload partition where the device must be mounted. File system specifications in the **-M** flag supersede file system specifications in the **image.data** file. There are 4 basic workload partition mount forms: **localfs** Disk-based file system (**vfs**=jfs or **vfs**=jfs2) to be created at the location that is specified by the value of the **directory** within the directory structure of the workload partition. If you specify a **dev** attribute, it denotes an existing logical volume in the global environment, which is to be used to host the file system. For localfs file systems, you must specify the **size** attribute. Other optional attributes, which are of the form *attr=value*, include those attributes in the following list: **logname** Specifies the log device to use for this file system. This attribute must be specified only if the default log device that the file system uses is insufficient. • For vfs=jfs2, the default is to use an inline log. • For vfs=jfs, the default is that the file system uses an existing log device if available. Otherwise, it creates one. When the **logname** attribute is being specified, make sure that the named log device exists. **mode** Specifies the octal permission mode to assign to the base directory of this file system. The default is 755. **size** Specifies the size of the file system that is created in a format acceptable to the **crfs** command. **vg** Specifies the volume group in which the file system (if no existing logical volume device is specified by using the **dev** attribute) is created. If you do not specify a value, the volume group that is specified in the **-g** flag is used. If you do not specify the **-g** flag, **rootvg** is assumed. Specifies other options to pass to the **crfs** command when the file system is being created. Options are passed directly to the **crfs** command so the value must be in the form that is required by the **crfs** command.

Item	Description
-M directory=*dir* [**vfs**=*type*] [**size**=*sizespec*] [**vg**=*volumegroup*] [**logname**=*loglv*] [**dev**=*devicepath*] [**host**=*RemoteHost*] [=] [**mountopts**=*mountopts*] (continued)	**Restriction:** Do not specify any options to the **crfs** command that correspond to the flags in the **mkwpar** command. The **mkwpar** command must not be specified by using the attribute because incorrect results might occur: • **-a logname**=*lvname* (logname) • **-a size**=*value* (size) • **-d** *device* (dev) • **-g** *volumegroup* (vg) • **-m** *mountpoint* (directory) • **-v** *vfstype* (vfs) For more information, see **crfs** documentation for further information about the **crfs** command. **mountopts** Specifies the mount options (corresponding to the "options" attribute in an /etc/filesystems stanza). If you do not specify a mount option, by default, no mount flags are used. Option values that you can specify correspond to the **-o** options of the **mount** command. **namefs** Specifies that the global directory that is specified by the **dev** attribute is mounted over the directory that is specified by the directory attribute in the file system structure of the workload partition. The only other attribute that is applicable to a **namefs** mount is **mountopts**. For the **namefs** type, you cannot map the /, /var, /opt, /usr, /tmp, or /proc file system of a workload partition with write privileges to a real /, /var, /opt, /usr, /tmp, or /proc file system. The **namefs** mount can also be used with rootvg workload partitions. In this case, the content of the namefs mount is not saved by using the **savewpar** command. **nfs** Specifies that the directory that is specified by the **dev** attribute on the system that is exported by the **host** attribute is mounted over the workload partition **directory**. The only other attribute that is applicable to a nfs mount is **mountopts**. **Requirement:** The global system and the workload partition must both have root permissions to the NFS device. You can give the global and the WPAR root permission to the NFS device, when you export the NFS mount, by specifying the root access for the host names of both the global system and the workload partition. When an NFS device is mounted, you cannot map the **/, /var, /opt**, or **/usr** file system of a workload partition with write privileges to a real **/, /var, /opt**, or **/usr** file system. **directory** Specifies that the directory that is specified by the **directory** attribute is added to the file system structure of the workload partition. No file system is created. Use this attribute to reduce the number of file systems to manage in a workload partition, such as by eliminating the separate file systems for **/tmp** and **/var**. Ensure that the size of the containing file system is adjusted accordingly. **Note:** A directory mount cannot be used for **/usr** or **/opt**.
-n *wparname*	Specifies the name for the workload partition to be created. You must specify a name, either by using the **-n** flag, or in a specification file by using the **-f** flag, unless the **-p** name or both **-w** and **-o** flags are used.

Item	Description

-N *attribute=value*	Specifies network configuration attributes. Separate the *attribute=value* pairs by blank spaces. You can specify more than one **-N** flag to configure multiple IP addresses. You must always specify the **address** or the **address6** attribute when you use the **-N** flag. Any other values that are not specified are taken from the settings of the global system. If you do not specify the **-N** flag, the **mkwpar** command attempts to discover an appropriate IP address for the workload partition. To do that, the **mkwpar** command performs the **gethostbyname** subroutine on the workload partition host name (specified with the **-h** flag). If no **-N** flag is specified and no host name is specified, the **mkwpar** command attempts to discover the IP address by performing the **gethostbyname** subroutine on the workload partition name (specified with the **-n** flag). If you can find an address on the same subnet as any global interface, use that interface settings with the resolved IP address to create the default network entry. You can specify the following attributes for the **-N** flag:
	• **interface**= *if* or **interface**=*namemappedif*
	• **address**=*A.B.C.D*
	• **netmask**=*A.B.C.D*
	• **broadcast**=*A.B.C.D*
	• **address6**=*S:T:U:V:W:X:Y:Z*
	• **prefixlen**=*n*
	
	The name-mapped interface is defined in the **/etc/wpars/devmap** file. You can specify the mapping between the name-mapped interface and the system interface as follows:
	`# The comments start with '#'`
	`# Each line contains a pair of name-mapped interface`
	`# and real interface separated by tab or blank spaces.`
	`foo en0`
	`goo en1`
	`soo en2`
	
	To define an IPv6 network configuration, specify the **-N** flag with the **address6**, **prefixlen**, and **interface** attributes. The **address6** attribute is a 128-bit address. The address is represented by eight 16-bit integers that are separated by colons. Each integer is represented by 4 hex digits. Leading zeros can be skipped, and consecutive null 16-bit integers can be replaced by two colons (one time per address). The **prefixlen** attribute is the number of high-order bits that are used to mask the IPv6 address and to comprise the prefix. The value of the **prefixlen** attribute ranges from 0 through 128. Each **-N** flag can accept either IPv4 attributes, or IPv6 attributes, but not both.

-o *outfile*	Indicates an output path and file name to which to write specification data. This specification file can then be used to create a workload partition later, by using the **-f** flag.
-O	This flag is used to force an existing volume group to be overwritten on a particular set of devices, specified with the **-D rootvg=yes** flag directive. If not specified, the overwrite value defaults to FALSE. This flag must be specified only once, as its setting is applied to all devices specified with the **-D rootvg=yes** flag directive.

Item	Description
-p [*name*]	Indicates that the workload partition's file systems exist and must be preserved, which means the root part must not be populated. You must specify the existing file systems to the **mkwpar** command in one of the following two ways: • Use the *name* parameter to specify an existing mount group in **/etc/filesystems**. Such a mount group usually exists because a previous workload partition was removed by using **rmwpar -p**. If you specify the *name* parameter, it cannot match the name of an existing workload partition on the system. If you specify the **-d** flag, the mount points of the file systems are adjusted accordingly. If you do not specify the **-d** flag, the base directory of the workload partition is determined based on the mount points that are associated with the discovered file systems. • If the file systems are not defined in **/etc/filesystems**, use the **-p** flag with the **-M** flag or mount stanzas in the specification file to define the attributes of the file systems. If you specify the **-p** flag with the *name* parameter, and no workload partition name is provided by using other means (for example, the **-n** flag or **general.name** in the specification file), you can also use the *name* parameter as the workload partition name. **Note:** The **-p** flag is mutually exclusive with the **-B** flag. The **-p** flag can be used to re-create a versioned workload partition. The workload partition's file systems must be preserved. To re-create such a versioned workload partition, the **-f infile** flag is included to provide a specification file that is saved from the original workload partition. The specification file is required to preserve the **ostype** value of the workload partition.
-P	Sets the root password for the workload partition. The **mkwpar** command prompts you for the password interactively.
-r	Duplicates the network name resolution configuration from the global system. The following files, if they exist, are copied into the workload partition: • **/etc/resolv.conf** • **/etc/hosts** • **/etc/netsvc.conf** • **/etc/irs.conf** • **/etc/networks** If the NSORDER environment variable is defined in the calling environment, it is added to the workload partition's `/etc/environment` file.
-R *attribute=value*	Allows specification of resource control attributes. You specify only one **-R** flag. Most resource controls are similar to those resource control attributes that are supported by Workload Manager. You can use the following attributes: **active={yes|no}** • **Active=no** means that the resource attributes are defined but the resource controls are not activated when the WPAR is started. • **Active=yes** means that the resource control attributes are activated when the WPAR starts. **Tip:** If this field is set to 'no', performance metrics such as processor and memory usage are not available by using such commands as **topas** and **wlmstat**, either inside and outside of the workload partition. **rset**=*rset* Configures the workload partition to use a resource set that was created by the **mkrset** command. **shares_CPU**=*n* Specifies the number of processor shares that are available to the workload partition. **CPU**=*m*%-*SM*%,*HM*% Specifies the percentage processor limits for the processes of the workload partition. **shares_memory**=*n* Specifies the number of memory shares that are available to the workload partition. **memory**=*m*%-*SM*%,*HM*% Specifies the percentage memory limits for the processes of the workload partition.

Item	Description
	procVirtMem=*n*[M | MB | G | GB | T | TB] Specifies the maximum amount of virtual memory that a single process can consume. Processes that exceed the specified limit are terminated. The valid units are megabytes (M or MB), gigabytes (G or GB), and terabytes (T or TB). The minimum limit that is allowed is 1 MB. The maximum limit that can be specified is 8796093022207M, 8589934591G, or 8388607T. If you set the value to -1 (no units), the limit is disabled. See Workload Manager limits File. **totalVirtMem**=*n*[M | MB | G | GB | T | TB] The maximum amount of virtual memory that can be consumed by the WPAR as a whole. Processes that cause the specified limit to be exceeded are terminated. The valid range and units are the same as **procVirtMem**. If you set the value to -1 (no units), the limit is disabled. See Workload Manager limits File. **totalProcesses**=*n* Specifies the total number of processes that are allowed in the workload partition. See Workload Manager limits File. **totalPTYs**=*n* Specifies the total number of pseudo terminals that are allowed in the workload partition. See pty Special File. **totalLargePages**=*n* Specifies the number of large pages that can be allowed for the workload partition. See Large Pages. **pct_msgIDs**=*n*% Specifies the percentage of the maximum number of message queue IDs of the system that are allowed in the workload partition. See Message Queue Kernel Services. **pct_semIDs**=*n*% Specifies the percentage of the maximum number of semaphore IDs of the system that are allowed in the workload partition. **pct_shmIDs**=*n*% Specifies the percentage of the maximum number of shared memory IDs of the system that are allowed in the workload partition. See Shared Memory. **pct_pinMem**=*n*% Specifies the percentage of the maximum pinned memory of the system that can be allocated to the workload partition. See Support for pinned memory. **totalThreads**=*n* Specifies the total number of threads that are allowed in the workload partition. See Workload Manager limits File.
rootvg=*yes* | *no*	Used to indicate whether the specified disk device is to be used as a rootvg WPAR device. If the **rootvg** option is not specified, the command takes the default of *no*.
-s	Starts the workload partition after it is created.

Item	Description
-S secfile = */path/to/secattrs* **privs[+|-]** = *list*	Configures the set of privileges that can be assigned to processes that are running in a system workload partition. You can provide privileges in a specification file (see the **-f** flag), in a separate security attributes file through **-S secfile=***/path/to/secattrs*, or on the command line by using the **-S privs=***list* flag. If you do not provide security attributes through one of these mechanisms, the **/etc/wpars/secattrs** file is used by default. When you use a separate security attributes file (either the default file or the file that is supplied through **-S secfile**), this file is read once when the workload partition is created to determine the privileges that are associated with the workload partition. Subsequent changes to the file have no effect on existing workload partitions. The default security attributes file */etc/wpars/secattrs* must not be modified directly as it might be overwritten in the future. If you use a base list of privileges from a specification file or security attributes file (including the default), individual privileges can be added to or removed from the list by specifying **-S privs+=***list*, **-S privs-=***list*, or both. Separate attributes must be separated by a blank space and must be unique, which means **secfile=, privs=, privs+=**, and **privs-=** cannot be specified more than once. Privileges must be comma-separated (without spaces) and must be unique. Attributes are processed in the following order regardless of the order that is specified in either the command line or the specification file: 1. The first attribute to be processed is the **privs** attribute without the **+** or **-** modifier. For example, **privs=PV_AZ_READ,PV_AZ_ADMIN**. If this attribute is found, no other attributes can be used. 2. The next attribute to be processed is the **secfile** attribute. See the security stanza of the Specification File Format for details on the format of this file. 3. If none of the attributes that are listed previously are specified, the **/etc/wpars/secattrs** file is used to populate the list of privileges. 4. The next attribute to be processed is an attribute with the **+** modifier. For example, **privs+=PV_DAC_UID,PV_AZ_ROOT**. This command adds the specified privileges to the list of privileges that are specified in the security file. 5. The final attribute to be processed is an attribute with the **-** modifier. For example, **privs-=PV_AZ_ROOT**. This command removes the specified privileges from the list of privileges that are specified in the security file. **Tip:** If you specify the **-S** flag on the command line, any security attributes in the specification file are ignored.

Item	Description
-u *userscript*	Specifies the path to a user script to be run by workload partition commands at various administration points. The parameter of the **-u** flag can be a string that is enclosed in quotation marks, including more arguments to be passed to the script. The first component of the parameter of the **-u** flag must be an absolute path to an existing executable file. The script is started in the following manner: `/path/to/userScript <action> <wparName>` The first argument indicates the administrative action that is being performed, as follows: **WPAR_LOAD** A script runs in the global environment after the kernel is configured, and before the tracked process is created. If the script returns a value other than zero, the workload partition cannot be started. **WPAR_START** A script runs in the global environment as soon as the workload partition becomes active. For system workload partitions, the script runs after the device configuration is complete. For application workload partitions, the script runs as soon as the tracked process is started. In the latter case, this code path can be run asynchronously by a dissociated process with its standard I/O streams closed or redirected. Internal messaging must be handled accordingly, and the script must account for the fact that short-lived workload partitions might be stopped or stopping at any point during the execution of the script. If the script returns a value other than zero, a warning is logged, but no other behavior changes. **WPAR_STOP** A script runs in the global environment after all workload partition processes finish before the kernel is unconfigured. **Note:** This code path can be started by a dissociated process with its standard I/O streams closed or redirected to SRC logs.If the script returns a value other than zero, a warning is logged, but no other behavior changes The second argument is the name of the workload partition. The script can use the **lswpar** command to obtain any other necessary configuration data.
-U [*Workload Partition UUID*]	Specifies the Workload Partition UUID. If you do not specify the value, the UUID is automatically generated for the corresponding Workload Partition.
-w	Writes the specification file only. Used with the **-o** flag, the **-w** flag causes the **mkwpar** command to quit after the new specification file is written, without actually creating the workload partition.
-W	Filters the bootset related information from the WPAR specification file. When the **-W** flag is used with the **-e** flag to generate a specification file from an existing WPAR, the resulting specification file does not include the bootset related attributes, such as bootset and bootlist.

Item	Description

-X [**exportfile**=*/path/to/file* |[**kext**=[*/path/to/extension* | *ALL*]] [**local**=*yes* | *no*] [**major**=*yes* | *no*]	Configures exporting kernel extensions that will be allowed to load inside a workload partition. You can specify more than one **-X** flag to allocate multiple kernel extensions. Separate the **attribute**=*value* by blank spaces. This flag is not valid for application workload partitions. You can specify the following attributes for the **-X** flag:
	
	exportfile=*/path/to/file*
	Specify a file containing valid extension stanzae that will be exported. An extension stanza should contain at least the **kext** attribute. The **local** and **major** attribute can also be specified in the stanza which are described below. The **exportfile** attribute is mutually exclusive with the **kext** attribute. It is also mutually exclusive with the local and major attribute because these can be specified for each extension stanza in the **exportfile**.
	
	This is a file that can be created by a user to use with **exportfile**=*/path/to/file* for **mkwpar** and **chwpar**. It can contain multiple extension stanzas. The **kext** attribute is required for each extension stanza. The local and major are optional as they both have default value **no**. The **exportfile** will look similar to the following.
	
	extension:
	`major = "yes"`
	`local = "no"`
	`kext = "/usr/lib/drivers/ldterm"`
	
	kext=*/path/to/extension*
	Specify a kernel extension that will be exported. This is a kernel extension located in the global system's filesystem. The keyword **ALL** can also be specified. This will allow a workload partition to load any extension. When **ALL** is specified, the **local** and **major** attributes are restricted to **local**=*yes* and **major**=*no*. Additional **-X** flags can be specified to overide the restricted local and major values. The **kext** attribute is mutually exclusive with the **-X exportfile** attribute.
	
	local=*yes* | *no*
	Specifying **local**=*yes* will make an instance of the kernel extension accessible to only the workload partition that is loading it. Specifying **local**=*no* will share the instance of the kernel extension loaded in the global system. By default, **local**=*no*.
	
	major=*yes* | *no*
	This attribute should only be used for kernel extensions that have an associated device major. By default, **major**=*no*.

-t	Copies the file systems from the rootvg volume groups from a system backup image specified by the **-B** flag or the global system.
-T [**preserve_private** ={**yes** | **no**}] [**preserve_wpars** ={**yes** | **no**}]	Controls behavior when copying the file systems from a rootvg volume group or system backup.
preserve_private ={**yes** | **no**}	Controls whether filesets that are designated as not visible within WPARs remain in a WPAR that is created by copying a rootvg volume group from a system backup or global system. The default value is no.
preserve_wpars ={**yes** | **no**}	Indicates whether the file systems associated with WPARs are available in the source system in a WPAR, which is created by copying a rootvg volume group from a global system. The default value is no.

Security

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

Examples

1. To create a workload partition called `roy`, enter the following command:
   ```
   mkwpar -n roy -N address=192.168.0.51
   ```
   All values that are not specified are generated or discovered from the global system settings.
2. To create a workload partition based on an existing specification file, enter the following command:
   ```
   mkwpar -f /tmp/wpar1.spec
   ```

3. To create a modified copy of a specification file with a new IP address, host name, and workload partition name (without creating a workload partition), enter the following command:

   ```
   mkwpar -f /tmp/wpar1.spec -N address=219.168.45.132 -h www.flowers.com -n wpar2
   -o /tmp/wpar2.spec -w
   ```

4. To create a specification file, which is based on an existing workload partition, enter the following command:

   ```
   mkwpar -e wpar1 -o /tmp/wpar2.spec -w
   ```

5. To recreate a workload partition that was previously removed with the **rmwpar -p** command, enter the following command:

   ```
   mkwpar -p wparname
   ```

6. To create a rootvg workload partition, enter the following command:

   ```
   mkwpar -n test -D devname=hdisk1 rootvg=yes -O
   ```

7. To create a rootvg workload partition called wpar1 with the storage device on an adapter, enter the following command (assuming that hdisk3 is attached to the adapter, fcs2):

   ```
   mkwpar -n wpar1 -D devname=fcs2 -D devname=hdisk3 rootvg=yes
   ```

8. To create a specification file from an existing workload partition, without including bootset related information, enter the following command:

   ```
   mkwpar -e <existing wparname> -W -w -o <path to spec file>
   ```

9. To create a WPAR with a default route, enter the following command:

   ```
   mkwpar -n wparB -N address=192.162.1.2 interface=en0 netmask=255.255.255.0 -i
   -I rtdest=0.0.0.0 rtgateway=192.162.1.1
   ```

10. To create a WPAR with its own routing table but no default route, enter the following command:

    ```
    mkwpar -n wparA -N address=192.152.1.2 interface=en0 netmask=255.255.255.0 -i
    ```

Files

Item	Description
/etc/wpars/devexports	Default device export control file for workload partitions.
/etc/wpars/secattrs	Default security file for workload partitions.
/usr/samples/wpars/sample.spec	An annotated workload partition specification file.

Related information:

rc.wpars command

rebootwpar command

startwpar command

Configuring WPAR-specific routing

---

mkwpardata Command

Purpose

Creates a file containing information about a workload partition for use by the **savewpar** and **restwpar** commands.

Syntax

mkwpardata [**-X**] [**-m**] *WparName*

Description

The **mkwpardata** command creates a file containing information about a workload partition (WPAR) for use by the **savewpar** and **restwpar** commands. The information includes the list of logical volumes, file systems and their sizes, the list of volume groups, and the WPAR name. The following files are

created:You can edit the information in the file before issuing the **savewpar** command.

Flags

Item	Description
-m	Creates map files that specify the mapping of the logical-to-physical partitions for each logical volume in the WPAR. 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. For example, for the **hd7** logical volume, the location of the map file is **/tmp/wpardata/***WparName***/hd7.map**. The MAPFILE field in the **image.data** file for the **hd7** logical volume is under the entry **MAPFILE=/tmp/wpardata/***WparName***/hd7.map**. The map files in the backup image are copied after the **image.data** file.
-X	Expands the **/tmp** file system if needed.

Parameters

Item	Description
WparName	Specifies the name of workload partition to back up.

Files

Item	Description
/tmp/wpardata/*WparName***/image.data**	Created for general and storage information about a WPAR. The *WparName* variable reflects the name of the WPAR. The **savewpar** command uses this file to create a backup image that can be used by the **restwpar** command to reinstall the WPAR. The **mkwpardata** command overwrites this file if it already exists. The **image.data** file is located in the **/tmp/wpardata/***WparName* directory, where *WparName* is the workload partition name.

Related information:

mkszfile command

restwpar command

savewpar command

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

Item	Description
-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

Item	Description
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

Item	Description
$TERM	Specifies the terminal names.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions for the **eqn** command and the **neqn** command.

Related reference:

"mmt Command"

Related information:

eqn command

neqn command

eqnchar command

mm Macro Package for the mm, mmt, nroff, and troff Commands

---

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

Item	Description
-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.

Item	Description
-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 reference:

"mm Command" on page 867

"mvt Command" on page 924

Related information:

grap command

mm Macro Package for the mm, mmt, nroff, and troff Commands

mmtu Command

Purpose

Displaying, adding, and deleting maximum transfer unit (MTU) values used for path MTU discovery.

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.

Note: The **-a** and **-d** flags used to modify the list of potential path MTU values are disallowed when executed within workload partitions.

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/sbin/mmtu	Contains the **mmtu** command.

Related information:

netstat command

no command

---

mobip6ctrl Command

Purpose

Configures and manages the mobile IPv6 home agent and correspondent node functionality.

Syntax

mobip6ctrl [**-c**] [**-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

Item	Description
-a *HomeAddress CareOfAddress MyAddress*	Adds this entry to the binding cache.
-b	Displays all binding cache entries.
-c	Compatibility option which enables the support of the mobiles implementing the draft #13 of the *Mobility support in IPv6* specification. Using this option, the home agent or correspondent node will accept the binding update messages sent using a Destination Option and using an Authentication Header (AH) to protect these packets with IPsec.
-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 reference:

"kmodctrl Command" on page 169

"mobip6reqd Daemon"

Related information:

Mobile IPv6

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 reference:

"mobip6ctrl Command" on page 872

"kmodctrl Command" on page 169

Related information:

Mobile IPv6

---

monacct Command

Purpose

Performs monthly or periodic accounting.

Syntax

/usr/sbin/acct/monacct [**-X**] [*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.

Flags

Item	Description
-X	Processes all available characters for each user name instead of truncating to the first 8 characters. The **-X** flag will also cause the **monacct** command to use the **/var/adm/acct/sumx** and **/var/adm/acct/fiscalx** directories instead of the **/var/adm/acct/sum** and **/var/adm/acct/fiscal** directories.

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.

Files

Item	Description
/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:

acctcms command

crontab command

acctmerg command

System accounting

---

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 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.

Flags

Item	Description
-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

Item	Description
/usr/lbin/tty/mon-cxma	Contains the **mon-cxma** command.
/tmp/mon-cxma.log	Contains the log file.

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

Item	Description
-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*.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
/usr/games	Contains the system's games.

Related information:

arithmetic command

number command

ttt command

turnon command

more Command

Purpose

Displays file contents one screen at a time.

Syntax

more [**-c**] [**-d**] [**-e**] [**-H**] [**-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.

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

Flags Description

Item	Description
-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.
-H	Disables the search pattern highlighting feature by default.
-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.

Flags Description

Item	Description
-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:

Item	Description
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.

more command

Item	Description
[*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.
H	Toggles the search pattern highlighting feature on or off.

Exit Status

This command returns the following exit values:

Exit Status

Item	Description
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

Files

Item	Description
/usr/share/lib/terminfo	Indicates the terminal information database.

Related reference:
"ksh Command" on page 173
Related information:
environment command
User environment
Understanding Locale Environment Variables

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 *output_defs_file* [**-s**] *inputfile ...*

mosy -x *output_desc_file* [**-o** *output_defs_file*] [**-s**] *inputfile ...*

mosy -c *output_c_file* [**-x** *output_desc_file*] [**-o** *output_defs_file*] [**-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 **mibII.my** format. Sample files are the **/usr/samples/snmpd/smi.my** and **/usr/samples/snmpd/mibII.my** files. See the **smi.my** and the **mibII.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 *output_defs_file* 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 **mibII.my** files are both specified as *inputfile* parameters. The **smi.my** file must precede the **mibII.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

Item	Description
-c *output_c_file*	Creates a C code file.
-o *output_defs_file*	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.
-x *output_desc_file*	Creates a description file in **mib.desc** file format.

Parameters

Item	Description
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/mibII.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.

4. To create a description file in **/tmp/smi.desc**, a C code file named **/tmp/smi.c**, and a Management Information Base (MIB) definition file named **/tmp/smi.defs**, enter:

   ```
   mosy  -x /tmp/smi.desc -c /tmp/smi.c -o /tmp/smi.defs -s smi.my mibII.my
   ```

Files

Item	Description
/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:

snmpinfo command

Understanding the Management Information Base (MIB)

Understanding Terminology Related to Management Information Base (MIB) Variables

mount Command

Purpose

Makes a file system available for use.

Syntax

mount [**-f**] [**-n** *node*] [**-o** *options*] [**-p**] [**-r**] [**-v** *vfsname*] [**-F** *AltFSfile*] [**-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/file systems** 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* or *node*:*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/file systems** 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/file systems** file.

The **/etc/file systems** 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 mount operations are referred to as automatic mount operations.

By default, the **mount** command runs the **wlmcntrl** command to refresh the current assignment rules in the kernel after mounting the file system. In some situations (such as when many file systems are mounted at once, or when a rule for an inaccessible remote mount is present in the workload manager configuration), calling **wlmcntrl** automatically after mount might be undesirable.

If you wish to override this behavior, set the environment variable *MOUNT_WLMCNTRL_SELFMANAGE* to any value. This will avoid calling the **wlmcntrl** command during the mount operation. You must manually run `wlmcntrl -u -d ""` to refresh the current assignment rules in the kernel. For more information, see **wlmcntrl** command.

Note:

1. If the **cdromd** CD and DVD **automount** daemon is enabled, those devices will be automatically mounted as specified in the **/etc/cdromd.conf** file. Use the **cdumount** or **cdeject** command to unmount an automatically mounted CD or DVD. Use **stopsrc -s cdromd** to disable the CD/DVD **automount** daemon.
2. For CacheFS, the remote file system that is to be cached locally must be exported such that the root ID of the local system is not remapped on the remote host to `nobody` (or the ID that the remote host uses as the anonymous user). For example, if host A were to export a file system **/F**, which would be mounted with CacheFS on host B, then the **/etc/exports** on host A would need to have an entry similar to:

   ```
   /F -rw,root=B
   or
   /F -ro,root=B
   ```

   depending on the mount options used for the local CacheFS mount.
3. Mounting a JFS file system on a read-only logical volume is not supported.
4. Mounting a JFS2 file system with EAv1 on Trusted AIX system converts the file system to EAv2.

Using mount on a JFS2 File System

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 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 file system to a system running an earlier release of the

operating system, the file system 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 file system should be returned to the system running the latter operating system release, and **fsck** should be run on the file system.

Flags

Item	Description
-a	Mounts all file systems in the **/etc/file systems** 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.
-F *AltFSfile*	Mounts on a file of an alternate file system, other than the **/etc/file systems** file.
-n *node*	Specifies the remote node that holds the directory to be mounted. The node can be specified as a colon-separated IPv6 address. If this is done with the `node:directory` format, the colon-separated IPv6 address must be enclosed in square brackets.
-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/file systems** file or any previous command-line options.
-t *type*	Mounts all stanzas in the **/etc/file systems** 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.

File System Specific Options

Item **Description**

-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:

atime Turns on access-time updates. If neither **atime** nor **noatime** is specified, **atime** is the default value.

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 file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. Using this option will prevent access in any manner other than CIO. It is impossible to use cached I/O on a file system mounted with the cio option. This means that mapping commands such as **mmap()** and **shmat()** will fail with EINVAL when used on any file in a file system mounted with the **cio** option. One side-effect of this is that it is impossible to run binaries out of a **cio** mounted file system, since the loader may use **mmap()**.
Note: When you mount the file system by using the **cio** option, all applications must manage the serialization of files. Quotas are not supported by the **cio** option because quotas have their own serialization code.

dev Specifies that you can open devices from this mount. If neither **dev** nor **nodev** is specified, **dev** is the default value.

dio Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call.
Note: Using the **-odio** or **-ocio** flags can help performance on certain workloads, but users should be aware that using these flags will prevent file caching for these file systems. Because readahead is disabled for these file systems, this may decrease performance for large sequential reads.

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 file system logging logical volume name where the following file-system operations are logged.

Item	Description
log=*NULL*	Turns off logging for JFS2 file systems. JFS2 depends on the log for metadata consistency, so if the system abnormally stops during JFS2 metadata operations, the file system cannot be recovered to a consistent state upon reboot. In these cases, the file system must be recreated. **Attention:** Because of the risk of data loss, use this flag with caution.
maxpout=*value*	Specifies the pageout level for files on this file system at which threads should be slept. If **maxpout** is specified, **minpout** must also be specified. Value must be non-negative and greater than **minpout**. The default is the kernel **maxpout** level.
minpout=*value*	Specifies the pageout level for files on this file system at which threads should be readied. If **minpout** is specified, **maxpout** must also be specified. Value must be non-negative. The default is the kernel **minpout** level.
maxRandW=*value*	Enables the random write behind function. To enable the random write behind function, you must specify a value greater than 0 for the **maxRandW** option and the **nRandCluster** option. The **maxRandW** option specifies a limit for the number of dirty pages per file that can remain in the system memory. The valid input for this option are value from 0 to 65536. If you do not specify a value, the global value is used. If you specify the string "global" for the value, the global tunable value is used.
noatime	Turns off access-time updates. Using this option can improve performance on file systems where a large number of files are read frequently and seldom updated. If you use the option, the last access time for a file cannot be determined. If neither **atime** nor **noatime** is specified, **atime** is the default value.
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. If neither **dev** nor **nodev** is specified, **dev** is the default value.
noguard	Mount the filesystem regardless of the current mountguard setting which would otherwise guard the filesystem against unsupported concurrent mounts in a PowerHA or other clustering environment. If mountguard is enabled by the **chfs** or **crfs** command, the filesystem cannot be mounted if it appears to be mounted on another node or system. Specifying the noguard option temporarily overrides the mountguard setting.
norbr	Mounts the file system without the release-behind-when-reading capability. If none of the release-behind options are specified, **norbrw** is the default value.
norbrw	Mounts the file system without both the release-behind-when-reading and release-behind-when-writing capabilities. If none of the release-behind options are specified, **norbrw** is the default value.
norbw	Mounts the file system without the release-behind-when-writing capability. If none of the release-behind options are specified, **norbrw** is the default value.
nRandCluster=*value*	Enables the random write behind function. To enable the random write behind function, you must specify a value greater than 0 for the **nRandCluster** option and the **maxRandW** option. The **nRandCluster** option specifies the number of clusters apart two consecutive writes must be in order to be random. A cluster is a consecutive range of disk blocks that are used up. The valid input for this option are value from 0 to 65536. If you do not specify a value, the global value is used. If you specify the string "global" for the value, the global tunable value is used.
rbr	Mount file system with the release-behind-when-reading capability. When sequential reading of a file in this file system is detected, the real memory pages used by the file will be released once the pages are copied to internal buffers. If none of the release-behind options are specified, **norbrw** is the default. **Note:** When **rbr** is specified, the **D_RB_READ** flag is ultimately set in the **_devflags** field in the **pdtentry** structure.

Item **Description**

rbw Mount file system with the release-behind-when-writing capability. When sequential writing of a file in this file system is detected, the real memory pages used by the file will be released once the pages written to disk. If none of the release-behind options are specified, **norbrw** is the default.
Note: When **rbw** is specified, the **D_RB_WRITE** flag is set.

rbrw Mount file system with both release-behind-when-reading and release-behind-when-writing capabilities. If none of the release-behind options are specified, **norbrw** is the default.
Note: If **rbrw** is specified, both the **D_RB_READ** and the **D_RB_WRITE** flags are set.

remount
Changes the mount options of a mounted file system. For JFS2 file systems, you can specify the following mount options with the remount option to change the settings of a mounted file system. For any mount options not specified, no change is made to the current corresponding settings of the file system.

atime, **noatime**; **dev**, **nodev**; **maxpout**, **minpout**; **rbr**, **norbr**; **rbw**, **norbw**; **rbrw**, **norbrw**, **rw**, **ro**, **rox**; **suid**, **nosuid**.
Note:

1. External-snapshot mounted file systems cannot be remounted to read-write file systems.
2. You cannot use the **rw** and **ro** remount options on a file system that is managed by data management application programming interface (DMAPI).

For NFS, there are three types of mount requests.

duplicate mount
If the node, object, mount point, and the options that are specified in the **mount** command are the same as those for an existing mount, the **mount** command returns information about a successful mount, but a new mount is not created.

new mount
If the remount option is not specified, the **mount** command creates a new mount. If the node, object, mount point, or the constant options that are specified in the **mount** command are different than those for the existing mounts, the **mount** command fails if the remount option is specified.

remount
If the node, object, and mount point are the same as those for a top-most mount, but the remount options are different, the remount operation modifies the mount options of an existing mount. In this case, NFS performs the remount operation.

A top-most mount does not have another mount on top of it. For remount requests, the following options can be modified: **acdirmax**, **acdirmin**, **acregmax**, **acregmin**, **actimeo**, **fastattr**, **grpid**, **hard**, **intr**, **noac**, **nocto**, **nodev**, **nointr**, **nosuid**, **posix**, **retrans**, **ro**, **rsize**, **rw**, **secure**, **sec**, **soft**, **timeo**, **wsize**, **biods**, **extraattr**, **nodircache**, **prefer**, **otwattr**, **maxgroups**, and **proto**. Other options are classified as constant options.

ro Specifies that the mounted file is read-only, regardless of its previous option specification in the **/etc/file systems** file or any previous command-line options. The default value is **rw**.

rw Specifies that the mounted file is read/write accessible, regardless of its previous option specification in the **/etc/file systems** file or any previous command-line options. The default value is **rw**.

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. The *snapshot* parameter specifies the name of an internal snapshot if the *snapshot* parameter does not included a forward slash (/), that is, no path information.

suid Specifies that execution of **setuid** and **setgid** programs by way of this mount is allowed. If neither **suid** nor **nosuid** is specified, **suid** is the default value.

Item	Description
	syncIntv=value Enables the **syncIntv** option if a value greater than 0 is specified and if the value of the **j2_syncByVFS** tunable is greater than 0. The valid values for this option are in the range of 0 to 86400 seconds.
	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 default is bin.
	WBCPages=*value* Enables the sequential write behind function if the value is greater than 0. You can use this option to specify the number of pages that you can schedule for write function at one time. The valid input for this option are value from 0 to 65536. If you do not specify a value, the global value is used. If you specify the string "global" option for the value, the global tunable value is used.
	wrkgrp=workgroup Specifies the workgroup that the SMB server belongs.

NFS Specific Options

Item	Description
-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**.
	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 30 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**.
	biods=*n* Sets the maximum number of **biod** threads that perform asynchronous I/O RPC requests for an NFS mount. The maximum value that can be set is 128. Values greater than 128 are limited to 128 within the NFS client. The NFS client dynamically manages the number of running **biod** threads up to the maximum based on activity. The default maximums for the different NFS protocols are 7 for NFS version 2 and 32 for NFS version 3 and NFS version 4. These defaults are subject to change in future releases.
	cio Specifies the file system to be mounted for concurrent readers and writers. I/O on files in this file system will behave as if they had been opened with **O_CIO** specified in the **open()** system call. Using this option will prevent access in any manner other than CIO. It is impossible to use cached I/O on a file system mounted with the cio option. This means that mapping commands such as **mmap()** and **shmat()** will fail with EINVAL when used on any file in a file system mounted with the **cio** option. One side-effect of this is that it is impossible to run binaries out of a **cio** mounted file system, since the loader may use **mmap()**. **Note:** When you mount the file system by using the **cio** option, all applications must manage the serialization of files. Quotas are not supported by the **cio** option because quotas have their own serialization code.

Item	Description
cior	Specifies to allow read-only files to open in the file system. I/O on files in this file system will behave as if they had been opened with **O_CIO | O_CIOR** specified in the **open()** system call. Using this option will prevent access in any manner other than **O_CIO | O_CIOR** and read-only. An attempt to open with **O_CIO** only will also fail. This option can only be used in conjunction with **cio**.
dio	Specifies that I/O on the file system will behave as if all the files had been opened with **O_DIRECT** specified in the **open()** system call. **Note:** Using the **-odio** or **-ocio** flags can help performance on certain workloads, but users should be aware that using these flags will prevent file caching for these file systems. Because readahead is disabled for these file systems, this may decrease performance for large sequential reads.
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.
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**.
nointr	Specifies no keyboard interrupts allowed on hard mounts.
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.
proto=[udp|tcp]	Specifies the transport protocol. The default is **tcp**. Use the **proto=[udp|tcp]** option to override the default. **proto=udp** cannot be specified if **vers=4**.

Item	Description
retrans=*n*	Sets the number of NFS transmissions to *n*. The default value is 5. The **retrans** setting determines how many times the NFS client retransmits a given UDP RPC request to an NFS server for file system operations. The **retrans** setting is not used during communication with the NFS server **rpc.mountd** service when processing NFS version 2 and 3 mounts. Retries to **rpc.mountd** are controlled with the **retry mount** option.
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. Beginning with AIX Version 6.1, the default value is 64 KB and the maximum value is 512 KB when using Version 3 and Version 4 of the NFS protocol.
secure	Specifies that the **mount** command uses Data Encryption Standard (DES) for NFS transactions. Data Encryption Standard (DES) is not supported in NFS Version 4, use *krb5* instead.
sec=*flavor***[:***flavor***...]**	Specifies a list of security methods that may be used to access files under the mount point. Allowable flavor values are: **sys** UNIX authentication. This is the default method. **dh** DES authentication. Data Encryption Standard (DES) is not supported in NFS Version 4, use *krb5* instead. **krb5** Kerberos. Authentication only. **krb5i** Kerberos. Authentication and integrity. **krb5p** Kerberos. Authentication, integrity, and privacy. The **secure** option may be specified, but not in conjunction with a **sec** option. The **secure** option is deprecated and may be eliminated in a future release. Use **sec=dh** instead.
sec=[*flavor1:...:flavorn***]**	The **sec** option specifies the security flavor list for the NFS mount. The available flavors are **des**, **unix**, **sys**, **krb5**, **krb5i**, and **krb5p**. This option only applies to AIX 5.3 or later.
shortdev	Specifies that you are mounting a file system from a host that does not support 32-bit device special files.
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, the default timeout is 100, which equals 10 seconds. For UDP mounts, the 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 timeout value is updated. The **timeo** option does not apply to communication from the NFS client to the **rpc.mountd** service on NFS servers. A timeout of 30 seconds is used when making calls to **rpc.mountd**.
vers=[2|3|4]	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|4]** option to select the NFS version. By default, the NFS mount will never use NFS Version 4 unless specified. The **vers=4** only applies to AIX 5.3 or later.
wsize=*n*	Sets the write buffer size to *n* bytes. Beginning with AIX Version 6.1, the default value is 64 KB and the maximum value is 512 KB when using Version 3 and Version 4 of the NFS protocol.

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.

Item	Description
-o	Specifies options. **acdirmax=***n* Specifies that cached attributes are held for no more than *n* seconds after directory update. Before *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 60 seconds. **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. **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. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option. **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.

Item	Description
	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. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option.
	purge Purges any cached information for the specified file system. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option.
	rw | ro Read-write (default) or read-only.
	suid | nosuid Allows (default) or disallows set-uid execution
	write-around | non-shared Writes 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. **Note:** If this option is used the first time a specific CacheFS is mounted, then the option must also be specified for subsequent mounts. There is state information stored in the cache control files that enforces consistent use of this option.
	mfsid Turns on global view. In NFS v4 system, you can traverse through the exported namespace on the server side. You need to specify this option to go over the file system. **Restriction: mfsid** is an option if the backend file system for CacheFS is NFS v4.
-V	Mounts a CacheFS file system.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To list the mounted file systems, enter the following command:

   ```
   mount
   ```

   This command produces output similar to the following:

   ```
   node   mounted          mounted over  vfs    date              options
   ----   -------          ------------ ---  ------------   -------------------
          /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 the following command:

   ```
   mount all
   ```

   This command sequence mounts all standard file systems in the **/etc/file systems** file marked by the **mount=true** attribute.

3. To mount a remote directory, enter the following command:

   ```
   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/file systems** file with a specific type, enter the following command:

   ```
   mount -t remote
   ```

   This command sequence mounts all files or directories in the **/etc/file systems** 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 the following command:

   ```
   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, enter the following command:

   ```
   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, enter the following command:

   ```
   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 file system, enter the following command:

   ```
   mount -v cifs -n pezman/user1/pass1 -o uid=201,fmode=750 /home /mnt
   ```
9. To remount the mounted read-only JFS2 file system to a read-write file system, enter the following command:

   ```
   mount –o remount,rw fsname
   ```
10. To mount all on a file **/tmp/fs1** of an alternate file system, enter the following command:

    ```
    mount -F /tmp/fs1 all
    ```

Files

Item	Description
/etc/file systems	Lists the known file systems and defines their characteristics.
/etc/vfs	Contains descriptions of virtual-file-system types.

Related information:

cdumount command

umount command

mount subroutine

/etc/filesystems file

mountd Daemon

Purpose

Answers requests from clients for file system mounts.

Syntax

/usr/sbin/rpc.mountd [**-n**] [**-N**] [**-x**][**-r**]

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

Item	Description
-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.
-x	Allows mount request without reverse lookup check.
-r	Turn off the **/etc/rmtab** file updates.

Files

Item	Description
/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:

showmount command

nfsd command

How to Mount a NFS File System Explicitly

How to Mount a File System Using Secure NFS

---

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

Item	Description
Device_Name	The name of the MPOA Client device name, for example, *mpc0*.

Flags

Item	Description
-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 reference:

"lecstat Command" on page 216

Related information:

entstat command

tokstat command

---

mpio_get_config Command

Purpose

Displays information about the DS3000 or the DS4000® subsystem that is based on multiple path I/O (MPIO) and the hdisks that are associated with the subsystem.

Syntax

mpio_get_config [**-v**] [**-a**] **-A** | **-l** <*hdisk#*> | **-?** | **-h**

Description

The **mpio_get_config** command displays information about the MPIO-based DS3000 or DS4000 subsystem and the hdisks that are associated with the subsystem.

Specifically, the command displays the information about the subsystem, including the assigned name of the subsystem, the worldwide name of the subsystem, and a list of hdisks in the **Available** state that are associated with the subsystem.

The following information about the **hdisk#** is displayed:

- hdisk name
- LUN number
- Current ownership
- Preferred path
- Adapter information
- User-assigned label for the volume

Flags

Item	Description
-A	Lists information for all attached subsystems.
-l	Lists information for the subsystem that includes the hdisk that is specified by the *hdisk#* parameter.
-v	Lists additional information about the controller and partition.
-a	Lists the adapter information.

Parameters

Item	Description
hdisk#	Specifies the hdisk name.

Examples

1. To display information about the subsystem that **hdisk11** is a member of, enter the following command:

   ```
   mpio_get_config -l hdisk11
   ```

   The system displays a message similar to the following message:

   ```
   Storage Subsystem Name = 'Twister'

   hdisk#           LUN #   Ownership          User Label
   hdisk11              0   A (preferred)      1_disk_0
   hdisk12              1   A (preferred)      1_disk_1
   hdisk13              2   A (preferred)      1_disk_2
   hdisk14              3   A (preferred)      1_disk_3
   ```

2. To display the information about the subsystem that **hdisk11** is a member of, along with the adapter information, enter the following command:

   ```
   mpio_get_config -a -l hdisk11
   ```

   The system displays a message similar to the following message:

   ```
   Storage Subsystem Name = 'Twister'
   hdisk#           LUN #   Ownership           Adapter  User Label
   hdisk11              0   A (preferred)        fscsi0  1_disk_0
   hdisk12              1   A (preferred)      Inactive  1_disk_1
   hdisk13              2   A (preferred)      Inactive  1_disk_2
   hdisk14              3   A (preferred)      Inactive  1_disk_3
   ```

3. To display information about all the attached subsystems, enter the following command:

   ```
   mpio_get_config -A
   ```

   The system displays a message similar to the following message:

   ```
   Storage Subsystem worldwide name: 60ab80026982e000045f255d7
   Storage Subsystem Name = 'Twister'

   hdisk#           LUN #   Ownership          User Label
   hdisk11              0   A (preferred)      1_disk_0
   hdisk12              1   A (preferred)      1_disk_1
   hdisk13              2   A (preferred)      1_disk_2
   hdisk14              3   A (preferred)      1_disk_3
   ```

4. To display additional information about all the attached subsystems, enter the following command:

   ```
   mpio_get_config -A -v
   ```

   The system displays a message similar to the following message:

   ```
   Frame id 0:
   Storage Subsystem worldwide name: 60ab80026982e000045f255d7
   Controller count: 2
   Partition count: 1
   Partition 0:
   ```

```
Storage Subsystem Name = 'Twister'
hdisk#            LUN #   Ownership          User Label
hdisk11               0   A (preferred)      1_disk_0
hdisk12               1   A (preferred)      1_disk_1
hdisk13               2   A (preferred)      1_disk_2
hdisk14               3   A (preferred)      1_disk_3
```

Related reference:

"manage_disk_drivers Command" on page 636

Related information:

Multiple Path I/O

---

mpstat Command

Purpose

Collects and displays performance statistics for all logical processors in the system.

Syntax

mpstat [{ **-d** | **-i** | **-s** | **-a** | **-h** | **-v** }] [**-w**][**-O***Options*] [**-@** *wparname*] [*interval* [*count*]]

mpstat [**-X** [-o *filename*]] [interval[count]]

Restriction: The *wparname* parameter is restricted to use inside workload partitions.

Description

The **mpstat** command collects and displays performance statistics for all logical processors in the system. Users can define both, the number of times the statistics are displayed, and the interval at which the data is updated.

The *interval* parameter specifies the amount of time in seconds between each report. If you do not specify the *interval* parameter, the **mpstat** command generates a single report that contains statistics for the time since system startup and then exits. You can specify the *count* parameter only with the *interval* parameter. If you specify the *count* parameter, its value determines the number of reports that are generated and the number of seconds apart. If you specify the *interval* parameter without the *count* parameter, reports are continuously generated. Do not specify a value of zero to the *count* parameter.

The **mpstat** command with no options generates a single report that contains the performance statistics for all logical processors since boot time.

When the **mpstat** command is invoked, it displays two sections of statistics. The first section displays the System Configuration, which is displayed when the command starts and whenever there is a change in the system configuration. The second section displays the Utilization Statistics which are displayed in intervals and at any time the values of these metrics are deltas from previous interval.

The following information is displayed in the system configuration section:

lcpu Indicates the number of online logical processors.

ent Indicates the entitled processing capacity in processor units. This information is displayed only when the partition type is shared.

mode Indicates whether the partition processor capacity is capped or uncapped allowing it to consume idle cycles from the shared pool. Dedicated LPAR is capped or donating.

rset Indicates the resource-set type (regular or exclusive) that is associated with the WPAR. This information is displayed only when there is a resource set that is associated with the WPAR.

The performance statistics displayed by **mpstat** are listed below:

CPU (All flags) Logical processor ID.

Note: The logical processor ID that is associated with the resource set of a WPAR is prefixed by an asterisk (*) when you run the **mpstat** command inside a WPAR with the **-s** or **-@** flag.

min (Default, **-a** flag) Minor page faults (page faults with no IO).

maj (Default, **-a** flag) Major page faults (page faults with disk IO).

mpcs (**-a**, **-i** flag) Number of `mpc` send interrupts.

mpcr (**-a**, **-i** flag) Number of `mpc` receive interrupts.

mpc (Only default) Total number of inter-processor calls .

dev (**-a**, **-i** flag) Number of device interrupts.

soft (**-a**, **-i** flag) Number of software interrupts.

dec (**-a**, **-i** flag) Number of decrementer interrupts.

ph (**-a**, **-i** flag) Number of phantom interrupts.

int (Only default) Total number of interrupts.

cs (Default, **-a** flag) Total number of context switches.

ics (Default, **-a** flag) Total number of involuntary context switches.

bound (**-a**, **-d** flag) Total number of threads that are bound.

rq (Default, **-a**, **-d** flag) Run queue size.

push (**-a**, **-d** flag) Number of migrations due to starvation load balancing .

S3pull
(-a, -d flag) Number of migrations outside the scheduling affinity domain 3 due to idle stealing.

S3grd (-a, -d flag) Number of dispatches from global runqueue, outside the scheduling affinity domain 3.

mig (Only default) Total number of thread migrations (to another logical processor).

S0rd (**-a**, **-d** flag) The percentage of thread redispatches within the same logical processor with scheduling affinity domain 0.

S1rd (**-a**, **-d** flag) The percentage of thread redispatches within the same physical processor or core with scheduling affinity domain 1.

S2rd (**-a**, **-d** flag) The percentage of thread redispatches within the same chip set, but not within the same processor core with scheduling affinity domain 2.

S3rd (**-a**, **-d** flag) The percentage of thread redispatches within the same MCM (multiple chip module) , but not within the same chip set with scheduling affinity domain 3.

S4rd (**-a**, **-d** flag) The percentage of thread redispatches on different MCMs within the same CEC or Plane with scheduling affinity domain 4.

S5rd (**-a**, **-d** flag) The percentage of thread redispatches on a different CEC or Plane with scheduling affinity domain 5.

S3hrd (**-a**, **-d** flag) The percentage of local thread dispatches on this logical processor.

S4hrd (**-a**, **-d** flag) The percentage of near thread dispatches on this logical processor.

S5hrd (**-a**, **-d** flag) The percentage of far thread dispatches on this logical processor.

lpa (Only default) Logical processor affinity. The percentage of logical processor re-dispatches within the scheduling affinity domain 3.

sysc (Default, **-a** flag) Number of system calls.

us (Default, **-a** flag, **-v** flag) The percentage of physical processor utilization that occurred while executing at the user level (application).

If the **-v** flag is used, then utilization is based on the virtual processor.

sy (Default, **-a** flag, **-v** flag) The percentage of physical processor utilization that occurred while executing at the system level (kernel).

If the **-v** flag is used, then utilization is based on the virtual processor.

wa (Default, **-a** flag, **-v** flag) The percentage of time that the logical processor was idle during which it had an outstanding disk I/O request.

If the **-v** flag is used, then utilization is based on the virtual processor.

id (Default, **-a** flag, **-v** flag) The percentage of time that the logical processor was idle and it did not have an outstanding disk I/O request.

If the **-v** flag is used, then utilization is based on the virtual processor.

pc (Default, **-a** flag, **-h** flag, **-v** flag) The number or fraction of physical processor consumed. It is displayed in both a shared partition and a dedicated partition. For the default flag in the dedicated partition, it is not displayed when both donation and simultaneous multithreading are disabled.

The **pc** of the `cpuid U` row represents the number of unused physical processors.

%ec (Default, **-a** flag) The percentage of entitled capacity consumed by the logical processor. The **%ec** of the ALL CPU row represents the percentage of entitled capacity consumed. Because the time base over which this data is computed can vary, the entitled capacity percentage can sometimes exceed 100%. This excess is noticeable only with small sampling intervals. The attribute is displayed only in a shared partition.

ilcs (**-a**, **-d**, **-h** flag) Number of involuntary logical processor context switches, displayed only in shared partition. For the **-h** and **-a** flags, it is also displayed in dedicated partition.

vlcs (**-a**, **-d**, **-h** flag) Number of voluntary logical processor context switches. Displayed only in shared partition. For the **-h** and **-a** flags, it is also displayed in dedicated partition.

lcs (Default) Total number of logical processor context switches. Displayed only in shared partition or when a dedicated partition is donating.

%idon

(**-a**, **-h** flag) Shows the percentage of physical processor utilization that occurs while explicitly donating idle cycles. Displayed only in dedicated partition that is donating.

%bdon

(**-a**, **-h** flag) Shows the percentage of physical processor utilization that occurs while donating busy cycles. Displayed only in dedicated partition that is donating.

%istol (**-a**, **-h** flag) Shows the percentage of physical processor utilization that occurs while the Hypervisor is stealing idle cycles. Displayed only in dedicated partition.

%bstol

(**-a**, **-h** flag) Shows the percentage of physical processor utilization that occurs while the Hypervisor is stealing busy cycles. Displayed only in dedicated partition.

%nsp (**-a**, **-d** flag) Shows the current average processor speed as a percentage of nominal speed. Displayed only if the hardware uses SPURR.

vcpu (**-v** flag) Shows the virtual processor id.

lcpu (**-v** flag) Shows the logical processor id.

pbusy (**-v** flag) Shows the percentage of physical time during which the physical processor is busy.

VTB (**-v** flag) Shows the time taken for a virtual processor in milliseconds.

The **mpstat** command displays all the above statistics for every logical processor in the partition. When running a shared partition, a special processor row with the `cpuid U` can be displayed when the entitled processing capacity has not entirely been consumed.

The **mpstat** command also displays a special processor row with the `cpuid ALL` which shows the partition-wide utilization. On that row, except for uncapped partitions with current physical processor consumption above their entitled capacity, the percentages are relative to the entitled processing capacity. For dedicated partitions, the entitled processing capacity is the number of physical processors. And for a WPAR , the processors present in the associated resource set, if any, are indicated by asterisks (*) only when the **-s** or **-@** flag is used.

When the **-s** flag is specified, the **mpstat** command reports simultaneous multithreading utilization, if it is enabled. This report displays the virtual processor engines utilization and utilization of each thread (logical processor) associated with the virtual processor engine.

If **mpstat** is running in a dedicated partition and simultaneous multithreading is enabled, then only the thread (logical processor) utilization is displayed.

If **mpstat** is running on an interval based mode, then it would be average value calculated per second.

Flags

Item	Description
-a	Displays all the statistics.
-d	Displays detailed affinity and migration statistics for AIX threads and dispatching statistics for logical processors.
-i	Displays detailed interrupts statistics.
-s	Displays simultaneous multithreading threads utilization, this flag is available only when **mpstat** runs in a simultaneous multithreading enabled partition.
-h	Displays pc and processor switches, with stolen and donation statistics for dedicated partitions.
-w	Displays wide column output, switches to wide output mode. Default is 80 column output mode.
@ *wparname*	Displays the statistics for the specified WPAR.
-O*Options*	Specifies the command option. **-O options=value...** Following are the supported options: • **sortcolumn** = Name of the metrics in the **mpstat** command output • **sortorder** = [asc|desc] • **topcount** = Number of CPUs to be displayed in the **mpstat** command sorted output
-X	Generates the XML output. The default file name is **mpstat_DDMMYYHHMM.xml** unless you specify a different file name by using with the **–o** option.
-o	Specifies the file name for the XML output.
-v	Displays utilization statistics at the virtual processor level. **Note:** The **-v** flag is available only for POWER8® processors, and later.

Note:

1. The **-a**, **-d**, and **-i** flags implicitly turn on wide-column output.
2. Inside a WPAR, the **-@** flag reports statistics of all processors.
3. Processor statistics that are displayed inside a WPAR is always system wide.
4. Only **-o** option is allowed with **-X** option.

Parameters

Item	Description
interval	Specifies the interval between the iterations. If *interval* is not specified, just one snapshot of metrics is displayed which actually reports the values from the time system is up. If *interval* is specified, the tool waits for that duration before printing the first set of data. Each set of data is followed by a separation line, a line with average values for each columns (except the processor, which is replaced by ALL), followed by an empty line.
count	Specifies number of iterations. If *interval* is specified and *count* is not specified then **mpstat** runs infinitely. *count* can not be specified without specifying *interval*.

Examples

1. To see the default set of utilization metrics, enter the following command:
   ```
   mpstat 1 1
   ```
2. To see the default set of utilization metrics in wide display mode, enter the following command:
   ```
   mpstat –w 1 1
   ```
3. To see the detailed dispatch & affinity metrics, enter the following command:
   ```
   mpstat –d 1 1
   ```
4. To see the detailed interrupts report, enter the following command:
   ```
   mpstat –i 1 1
   ```
5. To see all the statistics, enter the following command:
   ```
   mpstat –a 1 1
   ```
6. To see simultaneous multithreading utilization, enter the following command:
   ```
   mpstat –s 1 1
   ```
7. To see all the processor metrics of a WPAR, enter the following command:
   ```
   mpstat -@ wparname
   ```

   Note: To see all the processor metrics of a WPAR inside the WPAR, enter the following command:
   ```
   mpstat -@
   ```
8. To see the sorted output for the column **cs**, enter the following command:
   ```
   mpstat -d -O sortcolumn=cs
   ```
9. To see the list of the top 10 CPUs, enter the following command:
   ```
   mpstat -a -O sortcolumn=min,sortorder=desc,topcount=10
   ```
10. To see metrics based on the virtual processor, enter the following command:
    ```
    mpstat –v
    ```

Files

Item	Description
/usr/bin/mpstat	Contains the **mpstat** command.

Related reference:

"lparstat Command" on page 288
"iostat Command" on page 111

Related information:

vmstat command
sar command

mrouted Daemon

Purpose

Forwards a multicast datagram.

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

Item	Description
-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*
-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:

Item	Description
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

Item	Description
/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:

mrouted.conf command

---

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

Item	Description
-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

Item	Description
$HOME/.mh_profile	Contains the user's MH profile.
/etc/mh/mtstailor	Contains the MH tailor file.
/var/spool/Mail/$USER	Defines the location of the mail drop.
/usr/bin/msgchk	Contains the **msgchk** command.

Related reference:

"inc Command" on page 36

Related information:

mh_alias command

mh_profile command

Mail applications

---

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

Item	Description
-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:

Item	Description
`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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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

Item	Description
$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 reference:

"mhmail Command" on page 646

Related information:

vmh command

mh_alias command

Mail applications

---

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

Item	Description
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

Item	Description
-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

Item	Description
0	Indicates a successful completion.
>0	Indicates an error occurred.

Files

Item	Description
/dev/rmt/*n.n*	Specifies the raw streaming tape interface.
/usr/bin/mt	Contains the **mt** command file.

Related information:

tctl command

environment command

ioctl command

Tape drives

---

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

Item	Description
-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

Item	Description
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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

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 reference:

"mrouted Daemon" on page 906

Related information:

traceroute command

multibos Command

Purpose

Creates, updates, and manages multiple versions of the Base Operating System (BOS) on a **rootvg**.

Syntax

multibos -s [**-l** *device* {**-a** | **-f** *file* | **-b** *file* | **-x** *file*}] [**-e** *file*] [**-i** *file*] [**-L** *file*] [**-pntNX**]

multibos -c -l *device* {**-a** | **-f** *file* | **-b** *file* | **-x** *file*} [**-pnNX**]

multibos -m [**-pnX**]

multibos -u [**-pnX**]

multibos -B [**-ntX**]

multibos -S [**-nX**]

multibos -R [**-ptX**]

multibos -C [*VG name*]

multibos -s -M *file* [**-pntNX**]

Description

The **multibos** command allows the root level administrator to create multiple instances of AIX on the same **rootvg**. The **multibos** setup operation creates a standby Base Operating System (BOS) that boots from a distinct boot logical volume (BLV). This creates two bootable sets of BOS on a given **rootvg**. The administrator can boot from either instance of BOS by specifying the respective BLV as an argument to the **bootlist** command or using system firmware boot operations. Two bootable instances of BSO can be simultaneously maintained. The instance of BOS associated with the booted BLV is referred to as the *active* BOS. The instance of BOS associated with the BLV that has not been booted is referred to as the *standby* BOS. Currently, only two instances of BOS are supported per **rootvg**.

The **multibos** command allows the administrator to access, install maintenance and technology levels for, update, and customize the standby BOS either during setup or in subsequent customization operations. Installing maintenance and technology updates to the standby BOS does not change system files on the active BOS. This allows for concurrent update of the standby BOS, while the active BOS remains in production.

In addition, the **multibos** command copies or shares logical volumes and file systems. By default, the BOS file systems (currently **/**, **/usr**, **/var**, and **/opt**,) and the boot logical volume are copied. The administrator can make copies of additional BOS objects (using the **-L** flag).

All other file systems and logical volumes are shared between instances of BOS. Separate log device logical volumes (for example, those that are not contained within the file system) are not supported for copy and are shared.

In AIX 5L Version 5.3 with the 5300-09 Technology Level, you can populate the standby instance with a later version such as AIX Version 6.1 with the 6100-02 Technology Level. The function is done by creating a **mksysb** backup of a system at a later version and then using the backup to populate the standby instance. For example, system A is at a level of 5.3.9.0 and system B is at a level of 6.1.2.0. You can create a backup of system B using the `mksysb -M` command and use the **mksysb** backup to populate a standby instance of the operating system on system A.

Notes:

1. While the **multibos** command is running, ensure that system activity is minimal.
2. Any logical volume or file system attributes that are new to the higher level are note implemented when the standby instance is created because the operating system at a lower level has no knowledge of the attributes.
3. Do not keep both an AIX 6.1 instance and an AIX 5.3 instance for an extended period. You might not be able to switch between the instances because of incompatibilities. Commit to one of the instances and remove the other.

4. If your operating system is running with the logical volumes in the active BOS that have the multibos-created bos_hd* names, and no standby BOS in the rootvg directory, then a preservation or migration type of installation can occur starting with AIX 7200-00. The logical volumes that have the bos_hd* names on the system are bos_hd5, bos_hd4, bos_hd2, bos_hd9var, and bos_hd10opt. The operating system must not have hd5, hd4, hd2, hd9var, or hd10opt logical volumes. If you created the multibos instance from an mksysb image, which was created by using the **mksysb** command with the **-M** flag, the hd8 logical volume might also have been renamed to bos_hd8. You can check this prerequisite with the **lsvg -l rootvg** command. Always back up your system before migrating. Also, copy the /usr/lpp/bos/pre_migration file from the media or your network installation manager (NIM) spot of the level to which you are migrating, to the target system and execute the file on the target system to check for any migration warnings.

   Before you perform a migration or a preservation type of operating system installation in this environment, verify that the disk control block has a valid level for your rootvg. You can run the **/usr/lpp/bosinst/blvset -d /dev/hdiskN -g level** command, where hdiskN is the disk that contains the bos_hd5 logical volume. If this command returns 0.0 or an unexpected level, run the **bosboot -ad /dev/ipldevice** command to correct it, and rerun the **blvset** command to verify. It must return 6.1 or 7.1.
5. In addition to the flags mentioned in the syntax section, the **-V** flag performs the verify operation from the inittab during boot. It is important that you do not modify this entry. The verify operation enables the **multibos** utility to synchronize changes in logical volumes and file systems between the active and standby instances. This entry also synchronizes the ODM and devices on initial boot after a **mksysb** restore. Without this operation, both the active and standby instances could become inconsistent with normal file system and logical volume operations.

The file system types (JFS or JFS2) of the **mksysb** backup need to be the same as that of the system where the **multibos** command is to be run. For example, if the **/usr** file system is a JFS2 file system, the **/usr** file system on the **mksysb** backup needs to be a JFS2 file system.

A log is stored in the **/etc/multibos/logs/op.alog** file after you run the **multibos** command. You can view the log file using the **alog -f /etc/multibos/logs/op.alog -o** command.

Note: You can create a backup that contains both instances by first mounting the standby instance (using the **-m** flag), and then creating the backup. However, you can restore the backup onto a disk only by using the **alt_disk_mksysb** command.

Flags

Item	Description
-a	Specifies the **update_all** install option. Valid with setup and customization operation.
-B	Build boot image operation. The standby boot image is created and written to the standby BLV using the AIX **bosboot** command.
-b *file*	Specifies the install bundle to be installed during the setup or customization operation. The install bundle syntax should follow **geninstall** conventions.
-c	Performs a customized update of the software in standby BOS.
-C *VG name*	Allows you to vary on volume groups for syncing when it is appropriate, as the auto varied off volume groups are not varied on during the reboot, like the rootvg volume group, in order to be synced. The **multibos -C** *VG name* command should be used on auto varied off volume groups after the **multibos** command has created an alternative root volume group (operating system) on the disk and has been booted.
-e *file*	Lists active BOS files to be excluded during the setup operation in regular expression syntax.
-f *file*	Lists fixes (such as APARs) that are to be installed during the setup or customization operation. The syntax of the list follows **instfix** conventions.
-i *file*	Specifies optional **image.data** file to use instead of the default **image.data** file created from the current **rootvg**.
-L *file*	Specifies a file that contains a list of additional logical volumes to include in standby BOS.

Item	Description
-l *device*	Installs device or directory for software update during the setup or customization operation.
-m	Mounts standby BOS.
-M *file*	Specifies a file that contains a **mksysb** image. The **mksysb** image must have been created using the `mksysb -M` command beginning with AIX 6.1 with 6100-02.
-N	Skips boot image processing. This flag should only be used by experienced administrators that have a good understanding of the AIX boot process.
-n	Does not perform cleanup upon failure. This option is useful to retain **multibos** data after a failed operation.
-p	Performs a preview of the given operation. Valid with setup, remove, mount, unmount, and customization operations.
-R	Removes all standby BOS objects.
-S	Initiates an interactive shell with **chroot** access to the standby BOS file systems.
-s	Creates an instance of standby BOS.
-t	Prevents **multibos** from changing the bootlist.
-u	Unmounts standby BOS.
-x *file*	Runs the optional customization script before any other customized parameters, such as, update_all (**-a**), install bundle file (**-b**), and fix list file (**-f**). You must use a full pathname for the script.
-X	Allows for automatic file system expansion if space is needed to perform tasks related to **multibos**. It is recommended that all **multibos** operations are executed with this flag.

Exit Status

Item	Description
0	All the **multibos** command operations completed successfully.
>0	An error occurred.

Security

Only the root user can run the **multibos** command.

Examples

1. To perform a standby BOS setup operation preview, enter the following command:
   ```
   multibos -Xsp
   ```
2. To set up standby BOS, enter the following command:
   ```
   multibos -Xs
   ```
3. To set up standby BOS with optional **image.data** file **/tmp/image.data** and exclude list **/tmp/exclude.list**, enter the following command:
   ```
   multibos -Xs -i /tmp/image.data -e /tmp/exclude.list
   ```
4. To set up standby BOS and install additional software listed as bundle file **/tmp/bundle** and located in the images source **/images**, enter the following command:
   ```
   multibos -Xs -b /tmp/bundle -l /images
   ```
5. To execute a customization operation on standby BOS with the **update_all** install option, enter the following command:
   ```
   multibos -Xac -l /images
   ```
6. To mount all standby BOS file systems, enter the following command:
   ```
   multibos -Xm
   ```
7. To perform a standby BOS remove operation preview, enter the following command:
   ```
   multibos -RXp
   ```

8. To remove standby BOS, enter the following command:
   ```
   multibos -RX
   ```
9. To use an existing **mksysb** file **/backups/mksysb1** to populate the standby instance of rootvg, enter the following command:
   ```
   multibos -M /backups/mksysb1 -sX
   ```

Restrictions

- The **multibos** command is supported on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package and later.
- The current **rootvg** must have enough space for each BOS object copy. BOS object copies are placed on the same disk or disks as the original.
- The total number of copied logical volumes cannot exceed 128. The total number of copied logical volumes and shared logical volumes are subject to volume group limits.

Files

Item	Description
/usr/sbin/multibos	Contains the **multibos** command.
/etc/multibos	Contains **multibos** data and logs.

Related information:

multibos utility

---

mv Command

Purpose

Moves files.

Syntax

To Move and Rename a File

mv [**-d**] [**-e**] [**-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceFile ... TargetFile*

To Move and Rename a Directory

mv [**-d**] [**-e**] **-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceDirectory ... TargetDirectory*

To Move Files or Directories to a Directory Maintaining Original File Names

mv [**-d**] [**-e**] **-E**{**force**|**ignore**|**warn**}] [**-i** | **-f**] [**-I**] *SourceFile/SourceDirectory 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.

This warning message and prompt for input can be overridden by using the **-f** option.

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, the file mode, the extended attributes, and ACLs 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.

The **mv** command will not move an object if the object is exported as an NFS version 4 referral. The referral object is marked as busy and remains so until it is unexported.

Note: The I/O buffer size for the read and write system calls generated by this command can be configured by using the *AIX_STDBUFSZ* environment variable.

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.

Item	Description
-d	The source file is stored in decrypted (clear-text) format on target.
-e	The source file is stored in encrypted form, if the target file system is an Encrypted File System (EFS).
-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.
-I	Suppresses the warning message during ACL conversion.

The following table shows the encryption or decryption status of the target file under different conditions:

Explicit flag for the mv command	Source file	Target file system	Result
-e (encrypted)	Non-EFS	Non-EFS	Error
-e	Non-EFS	EFS	Encrypted file
-e	EFS	EFS	Encrypted file
-e	EFS	Non-EFS	Error
-d (decrypted)	Non-EFS	Non-EFS	Clear-text file
-d	Non-EFS	EFS	Clear-text file
-d	EFS	Non-EFS	Clear-text file
-d	EFS	EFS	Clear-text file
No explicit flag	Non-EFS	Non-EFS	Clear-text file
No explicit flag	Non-EFS	EFS	If the target file system is EFS activated, the target file is an encrypted file. Else, the target file is a clear-text file.
No explicit flag	EFS	EFS	Encrypted file
No explicit flag	EFS	Non-EFS	Error

Note: It is not permitted to overwrite an encrypted file with a plain-text file and vice versa unless you specify the **-f** flag. The encryption status of the target depends on the **-e** or **-d** flag, the encryption inheritance if you do not specify the **-e** or **-d** flag with the **-f** flag, and the encryption status of the source file if the encryption inheritance is not active.

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

Item	Description
0	All input files were moved successfully.
>0	An error occurred.

Files

Item	Description
/usr/bin/mv	Contains the **mv** command.

Related reference:

"ln Command" on page 239

Related information:

chmod command

rename subroutine

Input and output redirection

Miscellaneous tunable parameters

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.

The **mvdir** command will not rename a directory if the directory is exported for use by NFS version 4, or if the directory leads to a directory exported for use by NFS version 4. NFS version 4-exported directories and directories leading to NFS version 4-exported directories are marked as busy and remain so until unexported.

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

Item	Description
/usr/sbin/mvdir	Contains the **mvdir** command.

Related reference:

"mkdir Command" on page 703

"mv Command" on page 919

Related information:

Directories command

File and directory access modes

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. IPsec filter rules for this command can be configured using the **genfilt** command or IPsec smit (IP version 4 or IP version 6).

Flags

Item	Description
-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.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

---

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.

Item	Description
-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

Item	Description
TYPESETTER	Contains information about a particular printing device.

Files

Item	Description
/usr/share/lib/pub/eqnchar	Contains special character definitions.

Related information:

eqn command

pic command

tbl command

tc 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

Flags

Flag Description

Item	Description
-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*.

Flag Description

Item	Description
-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:

Windows Description

Item	Description
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.

Icon Box

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.

Icon Menu for Icon Box

Selection	Accelerator	Description
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:

Input focus

Button Action	Object	Function Description
Button 1 press	Window or window frame	Keyboard focus selection
Button 1 press	Icon	Keyboard focus selection

Function Description

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:

Windows Stacking function description

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:

Transient windows

Item	Description
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:

X Defaults description

Item	Description
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:

Component Appearance Resources

Item	Description
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

Component Appearance Resource Set

Name	Properties
background	**Class** **Background** **Value type** color **Default** varies[1]
backgroundPixmap	**Class** **BackgroundPixmap** **Value type** string[2] **Default** varies[1]
bottomShadowColor	**Class** **Foreground** **Value type** color **Default** varies[1]
bottomShadowPixmap	**Class** **BottomShadowPixmap** **Value type** string[2] **Default** varies[1]
fontList	**Class** **FontList** **Value type** string[3] **Default** "fixed"
foreground	**Class** **Foreground** **Value type** color **Default** varies[1]
saveUnder	**Class** **SaveUnder** **Value type** True of False **Default** False

Component Appearance Resource Set

Name	Properties
topShadowColor	**Class** **Background** **Value type** color **Default** varies[1]
topShadowPixmap	**Class** **TopShadowPixmap** **Value type** string[2] **Default** varies[1]
background	**Class** **Background** **Value type** color **Default** varies[1]
backgroundPixmap	**Class** **BackgroundPixmap** **Value type** string[2] **Default** varies[1]
bottomShadowColor	**Class** **Foreground** **Value type** color **Default** varies[1]
bottomShadowPixmap	**Class** **BottomShadowPixmap** **Value type** string[2] **Default** varies[1]
fontList	**Class** **FontList** **Value type** string[3] **Default** "fixed"
foreground	**Class** **Foreground** **Value type** color **Default** varies[1]
saveUnder	**Class** **SaveUnder** **Value type** True of False **Default** False
topShadowColor	**Class** **Background** **Value type** color **Default** varies[1]

Component Appearance Resource Set

Name	Properties
topShadowPixmap	**Class** **TopShadowPixmap** **Value type** string[2] **Default** varies[1]

Note:

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

Item	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.

Frame and Icon Component Resource Set

Name	Properties
activeBackground	**Class** **Background** **Value type** color **Default** varies[1]
activeBackground-Pixmap	**Class** **BackgroundPixmap** **Value type** string[2] **Default** varies[1]
activeBottomShadow-Color	**Class** **Foreground** **Value type** color **Default** varies[1]
activeBottomShadow-Pixmap	**Class** **BottomShadow-Pixmap** **Value type** string[2] **Default** varies[1]
activeForeground	**Class** **Foreground** **Value type** color **Default** varies[1]
activeTopShadowColor	**Class** **Background** **Value type** color **Default** varies[1]
activeTopShadowPixmap	**Class** **TopShadowPixmap** **Value type** string[2] **Default** varies[1]
activeBackground	**Class** **Background** **Value type** color **Default** varies[1]
activeBackgroundPixmap	**Class** **BackgroundPixmap** **Value type** string[2] **Default** varies[1]
activeBottomShadowColor	**Class** **Foreground** **Value type** color **Default** varies[1]

Frame and Icon Component Resource Set

Name	Properties
activeBottomShadowPixmap	**Class** **BottomShadowPixmap** **Value type** string[2] **Default** varies[1]
activeForeground	**Class** **Foreground** **Value type** color **Default** varies[1]
activeTopShadowColor	**Class** **Background** **Value type** color **Default** varies[1]
activeTopShadowPixmap	**Class** **TopShadowPixmap** **Value type** string[2] **Default** varies[1]

Note:

1. The default is chosen based on the visual type of the screen.
2. Image name.

Background

Item	Description
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.

General Appearance and Behavior Resource Set

Name	Properties
autoKeyFocus	**Class** **AutoKeyFocus** **Value type** True or False **Default** True
autoRaiseDelay	**Class** **AutoRaiseDelay** **Value type** millisec. **Default** 500
bitmapDirectory	**Class** **BitmapDirectory** **Value type** directory **Default** **/usr/include/X11/bitmaps**
buttonBindings	**Class** **ButtonBindings** **Value type** string **Default** "DefaultButton-Bindings"
cleanText	**Class** **CleanText** **Value type** True or False **Default** True
clientAutoPlace	**Class** **ClientAutoPlace** **Value type** True or False **Default** True
colormapFocusPolicy	**Class** **ColormapFocus-Policy** **Value type** string **Default** keyboard
configFile	**Class** **ConfigFile** **Value type** file **Default** **.mwmrc**

General Appearance and Behavior Resource Set

Name	Properties
defaultSetBehavior	**Class** **defaultSetBehavior** **Value type** True or False **Default** **True**
deiconifyKeyFocus	**Class** **DeiconifyKeyFocus** **Value type** True or False **Default** True
doubleClickTime	**Class** **DoubleClickTime** **Value type** milliseconds **Default** multiclick time
enableWarp	**Class** **enableWarp** **Value type** True or False **Default** True
enforceKeyFocus	**Class** **EnforceKeyFocus** **Value type** True or False **Default** True
fadeNormalIcon	**Class** **FadeNormalIcon** **Value type** True or False **Default** False
feedbackGeometry	**Class** **FeedbackGeometry** **Value type** string **Default** center on screen
frameBorderWidth	**Class** **FrameBorderWidth** **Value type** pixels **Default** varies
iconAutoPlace	**Class** **IconAutoPlace** **Value type** True or False **Default** True
iconBoxGeometry	**Class** **IconBoxGeometry** **Value type** string **Default** 6x1+0-0

General Appearance and Behavior Resource Set

Name	Properties
iconBoxName	**Class** **IconBoxName** **Value type** string **Default** iconbox
iconBoxSBDisplayPolicy	**Class** **IconBoxSBDisplayPolicy** **Value type** string **Default** all
iconBoxTitle	**Class** **IconBoxTitle** **Value type** XmString **Default** Icons
iconClick	**Class** **IconClick** **Value type** True or False **Default** True
iconDecoration	**Class** **IconDecoration** **Value type** string **Default** varies
iconImage-Maximum	**Class** **IconImage-Maximum** **Value type** width x height **Default** 50x50
iconImageMinimum	**Class** **IconImageMinimum** **Value type** width x height **Default** 16x16
iconPlacement	**Class** **IconPlacement** **Value type** string **Default** left bottom
iconPlacementMargin	**Class** **IconPlacementMargin** **Value type** pixels **Default** varies
interactivePlacement	**Class** **InteractivePlacement** **Value type** True or False **Default** False

General Appearance and Behavior Resource Set

Name	Properties
keyBindings	**Class** **KeyBindings** **Value type** string **Default** "DefaultKeyBindings"
keyboardFocusPolicy	**Class** **KeyboardFocusPolicy** **Value type** string **Default** explicit
limitResize	**Class** **LimitResize** **Value type** True or False **Default** True
lowerOnIconify	**Class** **LowerOnIconify** **Value type** True or False **Default** True

Maximum size

Item	Description
maximumMaximuSize	**Class** **MaximumMaximuSize** **Value type** width x height (pixels) **Default** 2X screen width & height
moveOpaque	**Class** **MoveOpaque** **Value type** True or False **Default** False
moveThreshold	**Class** **MoveThreshold** **Value type** pixels **Default** 4
multiScreen	**Class** **MultiScreen** **Value type** True or False **Default** False
passButtons	**Class** **PassButtons** **Value type** True or False **Default** False

Maximum size

Item	Description
PassSelectButton	**Class** **PassSelectButton** **Value type** True or False **Default** True
positionIsFrame	**Class** **PositionIsFrame** **Value type** True or False **Default** True
positionOnScreen	**Class** **PositionOnScreen** **Value type** True or False **Default** True
quitTimeout	**Class** **QuitTimeout** **Value type** milliseconds **Default** 1000
raiseKeyFocus	**Class** **RaiseKeyFocus** **Value type** True or False **Default** False
resizeBorderWidth	**Class** **ResizeBorderWidth** **Value type** pixels **Default** varies
resizeCursors	**Class** **ResizeCursors** **Value type** True or False **Default** True
screens	**Class** **Screens** **Value type** string **Default** varies
showFeedback	**Class** **ShowFeedback** **Value type** string **Default** all
startupKeyFocus	**Class** **StartupKeyFocus** **Value type** True or False **Default** True

Maximum size

Item	Description
transientDecoration	**Class** **TransientDecoration** **Value type** string **Default** menu title
transientFunctions	**Class** **TransientFunctions** **Value type** string **Default** -minimize -maximize
useIconBox	**Class** **UseIconBox** **Value type** True or False **Default** False
wMenuButtonClick	**Class** **WMenuButtonClick** **Value type** True or False **Default** True
wMenuButtonClick2	**Class** **WMenuButtonClick2** **Value type** True or False **Default** True

AutoKeyFocus

Item	Description
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**.

Item	Description
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**.
defaultSetBehavior (class **DefaultSetBehavior**)	Determines whether the **mwm** command automatically add key bindings to the **f.set_behavior** function (see .mwmrc file). If the value for the **defaultSetBehavior** resource is **True** (or On), regardless of the key bindings defined in the **.mwmrc** configuration file, `Alt Ctrl<key>1` and `Alt Shift Ctrl<Key>exclam` is bound to the **f.set_behavior** function. If the value for the **defaultSetBehavior** resource is **False** (or Off), the **f.set_behavior** function is bound to a key specified in the **.mwmrc** configuration file. If no key bindings are specified in the configuration file, the **mwm** command uses the default key binding. The default value for the **defaultSetBehavior** resource is **True** (or On).
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.

Item	Description
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.

Item	Description
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. The 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.

Key Bindings

Item	Description
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.

Item	Description
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.

Item	Description
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.

Client-Specific Resource Set

Name	Properties
clientDecoration	**Class** **ClientDecoration** **Value type** all **Default**
clientFunctions	**Class** **ClientFunctions** **Value type** string **Default** all
focusAutoRaise	**Class** **FocusAutoRaise** **Value type** True or False **Default** varies
iconImage	**Class** **IconImage** **Value type** pathname **Default** (image)
iconImageBackground	**Class** **Background** **Value type** color **Default** icon background
iconImageBottomShadowColor	**Class** **Foreground** **Value type** color **Default** icon bottom shadow
iconImageBottomShadowPixmap	**Class** **BottomShadowPixmap** **Value type** color **Default** icon bottom shadow pixmap
iconImageForeground	**Class** **Foreground** **Value type** color **Default** varies

Client-Specific Resource Set

Name	Properties
iconImageTopShadowColor	**Class** **Background** **Value type** color **Default** icon top shadow color
iconImageTopShadoPixmap	**Class** **TopShadowPixmap** **Value type** color **Default** icon top shadow pixmap
matteBackground	**Class** **Background** **Value type** color **Default** background
matteBottomShadowColor	**Class** **Foreground** **Value type** color **Default** bottom shadow color
matteBottomShadowPixmap	**Class** **BottomShadowPixmap** **Value type** color **Default** bottom shadow pixmap
matteForeground	**Class** **Foreground** **Value type** color **Default** foreground
matteTopShadowColor	**Class** **Background** **Value type** color **Default** top shadow color
matteTopShadowPixmap	**Class** **TopShadowPixmap** **Value type** color **Default** top shadow pixmap
matteWidth	**Class** **MatteWidth** **Value type** pixels **Default** 0
maximumClientSize	**Class** **MaximumClientSize** **Value type** width x height, vertical, horizontal **Default** fill the screen

Client-Specific Resource Set

Name	Properties
useClientIcon	**Class** **UseClientIcon** **Value type** True or False **Default** F
usePPosition	**Class** **UsePPosition** **Value type** string **Default** nonzero
windowMenu	**Class** **WindowMenu** **Value type** string **Default** DefaultWindowMenu

Client Decoration

Item	Description

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.

Item	Description
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**).

Client Decoration

Item	Description
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 *WidthxHeight*. 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).

Client Decoration

Item	Description
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:

Resource Description

Item	Description
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.

Item	Description
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.

Item	Description
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.

Item	Description
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.

Item	Description
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:

Function contexts

Item	Description
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

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
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

Normalize item description

Item	Description	
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

Normalize item description

Item	Description	
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.

Window Manager Event Specification

Item	Description
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 Event Name descriptions

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

Button Event Name descriptions

Button	Description
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:

Exit Status

Item	Description
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:

X command

Notices

This information was developed for products and services offered in the US.

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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 jurisdictions 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.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

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:

© Copyright IBM Corp. 2015, 2017

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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 actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as the customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at http://www.ibm.com/privacy and IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

INFINIBAND, InfiniBand Trade Association, and the INFINIBAND design marks are trademarks and/or service marks of the INFINIBAND Trade Association.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

Special characters

/etc/filesystems file
  listing entries
    using lsfs command 411
/etc/inittab file
  listing records
    using lsitab command 416
/etc/vfs file
  listing entries
    using lsvfs command 553

A

accounting system
  displaying information about last commands run 186
  displaying previous logins and logouts 184
  performing periodic accounting
    using monacct command 874
  updating last log in records 187
acct/* commands
  lastlogin 187
  monacct 874
appearance
  general
    using mwm command 936
Appearance resource set
  using mwm command 937
attributes
  displaying characteristics of
    using lsattr command 354
  displaying possible values of
    using lsattr command 354
  of exported symbols 208

B

behavior
  general
    using mwm command 936
Behavior resource set
  using mwm command 937
bibliography
  building inverted index for 42
  finding references in
    using lookbib command 261
binding
  button
    using mwm command 962
  key
    using mwm command 963
button
  binding
    using mwm command 962

C

C programming language
  checking source code for problems 226
  reformatting programs 38
CD-ROM
  reporting statistics 111
Client-Specific resource set
  using mwm command 949
code page
  converting one page encoding set to another 7
command
  mwm 925
commands
  event response resource manager (ERRM)
    logevent 248
  event-response resource manager (ERRM)
    lsevent 405
  ikedb 24
  installp 62
  inuwpar 89
  invscoutd 95
  join 143
  kdb 149
  keycomp 154
  keyenvoy 157
  ld 190
  logevent 248
  lphistory 310
  lppchk 315
  lpr 323
  ls 339
  lsassocmap 352
  lsattr 354, 806
  lsaudrec 359
  lscomg 377
  lscondition 380
  lscondresp 385
  lscons 391
  lsevent 405
  lslpclacl 424
  lslpcmd 429
  lslpp 432
  lslpracl 437
  lslpriacl 442
  lslprsacl 447
  lslv 452
  lsmcode 456
  lsresponse 493
  lsrpdomain 499
  lsrpnode 502
  lsrsrc 507
  lsrsrcassoc 513
  lssensor 532
  lsvsd 567
  lswpar 573
  mail 605
  Mail 605
  mailq 616
  mailx 605
  make 618
  mkboot 661
  mkcfsmnt 672
  mkcimreg 676
  mkclient 682
  mkcomg 686

© Copyright IBM Corp. 2015, 2017

commands *(continued)*
mkcondition 690
mkfs 716
mkitab 728
mklpcmd 736
mklv 739
mknfs 751
mknfsmnt 755
mkprtsv 775
mkresponse 785
mkrole 790
mkrpdomain 792
mkrsrc 802
mksensor 817
mkssys 825
mkstr 828
mkuser 839
mkuser.sys 843
mkvg 845
mkwpar 853
commands, mtrace 913
Component Appearance resource set
using mwm command 932
computer languages
C
lexical analyzer 218
CPU
statistics 111

D

daemons
lockd 245
monitord 877
mountd 896
mrouted 906
default window menu
using mwm command
using mwm command 927
description file
creating
using makedev command 629
using mwm command 955
device
adding to the system
using mkdev command 701
Device Configuration Database 472
listing acceptable devices from
using Isparent command 472
devices
displaying characteristics of
using lsdev command 396
diagnostic messages
issuing by optional programs 88
directories
creating 703
creating a hierarchy 705
moving 919, 922
renaming 922
directory
creating a lost and found
using mklost+found command 735
mounting
using mount command 885
disks
reporting statistics 111
displays
files
one screen at a time 878
listing currently available on the system
using lsdisp command 403
document
typesetting
using mmt command 869

E

ERRM
event information
logging 248
ERRM commands
logevent 248
lsevent 405
ERRM scripts
logevent 248
error messages
issuing by optional programs 88
event information
logging 248
event response resource manager (ERRM)
commands
logevent 248
event information
logging 248
scripts
logevent 248
event specification
using mwm command 961
event-response resource manager (ERRM)
commands
lsevent 405

F

file
creating a special
using mknod command 759
finding lines in a sorted
using look command 261
resource description
using mwm command 955
writing inode information 136
file system
constructing a prototype
using mkproto command 768
making available for use
using mount command 885
files
archive 190
creating links 239
displaying
one screen at a time 878
import and export 191
moving 919
filesets
installing, associated with keywords or fixes 73
fixes
installing filesets associated with 73
flags
ignored and unsupported 192
folder
incorporating new mail into 36

folders
printing full path names of
using mhpath command 647
font code
adding to the system
using mkfont command 714
font files
creating fonts.dir file
using mkfontdir command 714
font set
changing
using managefonts command 637
updating
using managefonts command 637
fonts
listing the fonts available for use
using lsfont command 410
Frame Component resource set
using mwm command 934

G

games
number-guessing game 878
garbage collection
using ld command 192
groups
creating new groups
using mkgroup command 719
displaying attributes of
using lsgroup command 412

H

help
using files, editors, and macros 214
hlpid 8
hlpindent 38
hlpiostat 111
hlpipcs 122
hlplearn 214
hlpleave 216
hlpline 224
hlplint 226
hlpln 239
hlplorder 265
hlpmkvgdata 850

I

I/O
reporting statistics 111
ibm3812 command 1
ibm3816 Command 2
ibm5587G command 4
IBM5587G printer
postprocessing troff command output for 4
ibm558H-T Command 3
ibstat command 5
icon
box menu
using mwm command 928
window menu
using mwm command 927
Icon Component resource set
using mwm command 934
iconifying window
using mwm command 927
iconv command 7
id command 8
idinstal 52
idnls 241
idprocess 121
ifconfig command 11
ikedb command 24
IMAP commands
imapd 30
imapds 31
imapd daemon 30
imapds daemon 31
impfilt command 32
import and export file format
-bI: and -bE: Flags 206
importing filter rules
from export files 32
importvg command 32
imptun command
adding exported tunnel definitions and filter rules 35
inc command 36
indent command 38
index
building inverted for bibliography 42
indxbib command 42
inetd daemon 43
infocmp command
managing terminfo descriptions 45
init command 48
inode numbers
information about 136
input extension records
listing
using listX11input command 233
input focus
keyboard
using mwm command 929
install applications
Installation Assistant 57
install command 52
install command (BSD) 59
install procedure
saving files changed during 84
install_all_updates command 54
install_assist command 57
install_mh command 58
Installation Assistant
starting
using the install_assist command 57
installing
programs in compatible package
using installp command 62
installing a command 52
installios command 60
installp command 62
installp format
creating software packages 725
instfix command 73
interprocess communication
removing identifiers 121
reporting status 122
inucp command 75
inudocm command 77
inurecv command
description of 80

inurecv command *(continued)*
performing archive operations for 81
performing restore operations for 81
inurest command 81
inurid command
removing installation information 83
inusave command
description of 84
performing archive operations for 81
performing restore operations for 81
inutoc command 87
inuumsg command 88
inuwpar command 89
inverted index
building for bibliography 42
invscoutd command 95
ioo command 100
iostat command 111
ipcrm command 121
ipcs command 122
ipreport command 126
iptrace daemon 129
ipv6policy command 132
Isallqdev command 350
isC2host command 133
isCChost command 134
Isconn command 390
Isdev command 396
isnstgtd command 135
ISO 2022 584
Isparent command
listing acceptable connection types from
using Isparent command 472
istat command 136

J

j2edlimit command 139
job control 141
jobs command 141
join command 143
joinvg command 146

K

kdb command 149
key
binding
using mwm command 963
keyadd command 152
keyboard maps
listing
using lskbd command 417
keycomp command 154
keydelete command 155
keyenvoy command 157
keylist command 157
keylogin command 159
keypasswd command 160
keyserv daemon 161
kill command 162
killall command 164
kmodctrl command 169
Korn shell
invoking 173
krlogind daemon
server function
providing 170
krshd daemon
server function
providing 172
ksh command 173
ksh93 command 176

L

labck 182
last command 184
lastcomm command 186
lastlogin command 187
ld command 190
archive files 190
attributes of exported symbols 208
garbage collection 192
ignored and unsupported flags 192
import and export file format 206
import and export files 191
libraries 191
linking mode 190
options (-bOptions) 195
processing 191
run-time linking 206
symbols 192
ldd command 211
ldedit 212
learn command 214
leave command 216
lex command
definitions 219
rules 220
specification file 219
lexical analyzer 218
libraries
checking for incompatibilities 226
using ld command 191
line command 224
line printer
generating a ripple pattern for a
using lptest command 337
sending requests to
using lp command 266
line printer spooling queue
removing jobs from
using lprm command 327
link command 225
link subroutine 225
linking mode
using ld command 190
links
hard 239
symbolic 239
lint command 226
listdgrp command 231
listing
software products
using lslpp command 432
listvgbackup command 231
listX11input command 233
Live Update
lvupdateInit command 591
lvupdateRegKE command 592
lvupdateRegScript command 593

Live Update *(continued)*
lvupdateSafeKE command 595
lvupdateSetProcs command 596
ln command 239
locale command
writes information about locales 241
localedef command
Processes locale and character map files 243
lock command 245
lockd daemon 245
locktrace command
controls kernel lock tracing 247
logevent command 248
logevent script 248
logform command 250
logger command 252
logical volume
adding mirrors to
using mklvcopy command 745
mirroring
using mirrorvg command 653
login command 253
login name
displaying the current process
using logname command 259
logins command 256
logname command 259
logoffs
displaying all previous 184
logout command 260
look command 261
lookbib command 261
loopmount command 262
loopumount command 264
lorder command 265
lp command 266
lpacl information 274
lpar_netboot command 285
lparstat 288
lpd command 299
lphistory command 310
lppchk command 315
lppmgr command 317
lpq command 319
lpr command 323
lprm command 327
lpstat command 331
lptest command 337
ls command 339
ls-secldapclntd 344
lsallq command 349
lsarm command 351
lsassocmap command 352
lsattr command 354
lsaudrec command 359
lsauthent command
authentication methods 365
lsC2admin command 366
lsCCadmin command 367
lscfg command 367
lscifscred command 370
lscifsmnt command 371
lsclass command 372
lscluster
configuration 374
lscomg command 377
lscondition command 380
lscondresp command 385
lscons command 391
lscore 393
lscosi command 394
lsdisp command 403
lsevent command 405
lsfilt command
listing filter rules 409
lsfont command 410
lsfs command 411
lsgroup command 412
lsiscsi command 414
lsitab command 416
lskbd command 417
lsldap command 419
lslicense command
fixed and floating licenses
listing number and status of 423
lslpclacl command 424
lslpcmd command 429
lslpp command 432
lslpracl command 437
lslpriacl command 442
lslprsacl command 447
lslv command 452
lsmaster command 455
lsmcode command 456
lsmksysb command 458
lsmp command 460
lsmpio command 461
lsnamsv command 466
lsnfsexp command 467
lsnfsmnt command 468
lsnim command 469
lsnlspath command 472
lspath command 474
lspprc command 485
lsprtsv command 480
lsps command 480
lspv command 482
lsque command 487
lsquedev
command 488
lsquedev command 488
lsresource command
displaying bus resources 490
lsresponse command 493
lsrole command 497
lsrpdomain command 499
lsrpnode command 502
lsrset command 506
lsrsrc command 507
lsrsrcassoc command 513
lssavevg command 522
lssavewpar command 524
lssec
listing attributes of
security stanza files 525
lssecattr command 528
lssecmode 531
lssensor command 532
lssrad command 541
lssrc command 542
lsts command 545
lstun command
listing tunnel definitions 547
lstxattr 548

lsuser command 550
lsusil command 553
lsvfs command 553
lsvg command 554
lsvgfs command 557
lsvirprt command 558
lsvmode command 561
  displaying current video mode 561
lsvpd command 562
lsvsd command 567
lswlmconf command 569
lswpar command 573
luit command 584
lvmo command 238, 585
lvmstat 588
lvupdateInit command 591
lvupdateRegKE command 592
lvupdateRegScript command 593
lvupdateSafeKE command 595
lvupdateSetProcs command 596

M

m4 command
  preprocessing files 599
mach command 602
machstat command 602
macref command
  producing cross-reference listing of 603
macro file 603
mail
  incorporating into a folder 36
  sending 646
mail command 605
Mail command 605
Mail commands
  mailq 616
  mailstats 617
mail traffic statistics
  displaying 617
mailbox directories, setting up 58
mailq command 616
mailstats command 617
mailx command 605
make command 618
makedbm command 625
makedepend command 626
makedev command 629
man command 631
man pages
  displaying information online 631
manage_disk_drivers command 636
managefonts command 637
mant command 639
manual
  typesetting pages of
    using mant command 639
mark command 640
memorandum macro 867
menu pane
  window manager
    using mwm command 963
mesg command 643
message facility commands
  mkcatdefs 664
message queues
  removing identifiers 121
message sequences
  creating
    using mark command 640
  displaying
    using mark command 640
  modifying
    using mark command 640
message source file
  preprocessing 664
messages
  checking for
    using msgchk command 909
  permitting
    using mesg command 643
  printing full path names of
    using mhpath command 647
  producing formatted lists of
    using mhl command 644
  refusing
    using mesg command 643
MH
  install_mh command 58
MH shell
  creating
    using msh command 910
mhl command 644
mhmail command 646
mhpath command 647
migratelp 649
migratepv command 650
migwpar command 652
mirror pools 460, 483
mirrorvg command 653
mirscan command 656
mkboot command 661
mkC2admin command 663
mkcatdefs command 664
mkCCadmin command 666
mkcd command 667
mkcfsmnt command 672
mkcifscred command 673
mkcifsmnt command 674
mkcimreg command 676
mkclass command 679
mkclient command 682
mkcluster
  cluster creation 683
mkcomg command 686
mkcondition command 690
mkcosi command 699
mkdev command 701
mkdir command 703
mkdirhier command 705
mkdom command 705
mkdvd command 707
mkfifo command
  making FIFO special files 712
mkfilt command
  activating or deactivating filter rules 713
mkfont command 714
mkfontdir command 714
mkfs command 716
mkgroup command
  description of 719
mkhosts command 723
mkiba command 724
mkinstallp command 725

mkiscsi command 727
mkitab command 728
mkkeyserv command 730
mklost+found command 735
mklpcmd command 736
mklv command 739
mklvcopy command 745
mkmaster command 748
mknamsv command 749
mknetid command 750
mknfs command 751
mknfsexp command 752
mknfsmnt command 755
mknfsproxy command 758
mknod command 759
mkpasswd command 762
mkpath command 764
mkprojldap command 766
mkproto command 768
mkprtldap command 772
mkprtsv command 775
mkps command 778
mkque command 780
mkquedev command 782
mkramdisk command 783
mkresponse command 785
mkrole command 790
mkrpdomain command 792
mkrset 801
mkrsrc command 802
mkrtc command 806
mksecldap command 809
mksecpki command 815
mksensor command 817
mkslave command 824
mkssys command 825
mkstr command 828
mksysb command 829
mkszfile command 833
mktcpip command 835
mkts command 837
mkuser command 839
mkuser.sys command 843
mkusil command 844
mkvg command 845
mkvgdata command 850
mkvirprt command 851
mkwpar command 853
mkwpardata command 866
mm command
  printing document formatted with
    using mm command 867
mmt command 869
mmtu command 871
mobip6ctrl commnad 872
mobip6reqd daemon 873
modify XCOFF header 212
monacct command 874
monitord daemon 877
mount command 885
mountd daemon 896
mounting
  automatic
    using mount command 886
mpio_get_config command 899
mpstat 901
mrouted daemon 906
msgchk command 909
msh command 910
mtrace command 913
multibos command 915
multicast path
  from a source to a receiver, printing a
    using mtrace command 913
MultiPath I/O
  lsmpio command 461
  lspath command 474
  mkpath command 764
multiple path I/O (MPIO) 636, 899
mvt command 924
MWM 925
mwm command 925

N

Network Install Manager 469
NFS commands
  mknfs 751
  mknfsexp 752
  mknfsmnt 755
NFS daemons
  lockd 245
  mountd 896
NIM commands
  lsnim 469
NIS commands
  keyenvoy 157
  keylogin 159
  lsmaster 455
  lsnfsexp 467
  lsnfsmnt 468
  makedbm 625
  mkclient 682
  mkkeyserv 730
  mkmaster 748
  mkslave 824
NIS daemons
  keyserv 161
NLSPATH,
  secure system configuration variable,
    showing the value of 472
notify object class (SRC)
  adding a notify method definition 761

O

object files
  reading one or more 265
objects
  shared 191
options (-bOptions)
  in ld command 195

P

packet tracing
  ipreport command 126
  iptrace daemon 129
paging space
  adding an additional
    using mkps command 778
  displaying the characteristics of
    using lsps command 480

password
reserving a terminal
using lock command 245
performing simply copy operations 75
physical volume
moving partitions from one volume to another volume
using migratepv command 650
plotter queue
listing device names with a queue
using lsallqdev command 350
policy
keyboard input
using mwm command 929
port
stopping all processes on a
using logout command 260
port 515
monitoring for print requests
using lpd command 299
print server
providing on a network
using lpd command 299
printer
line
sending requests to a 266
printer queue
adding to the system
using mkque command 780
using mkquedev command 782
listing device names with a queue
using lsallqdev command 350
printing a multicast path
from a source to a receiver
using mtrace command 913
process
initializing
using init command 48
using telinit command 48
stopping 162
process groups 141
processes
canceling all 164
stopping all port
using logout command 260
program products
listing optional
using lslpp command 432
programs
creating make file dependencies 626
installing compatible
using installp command 62
providing a C preprocessor interface to the make command 27

Q

queue
displaying the stanza name of
using lsque command 487

R

RAM disk 783
read operations
copying from standard input 224
remount option 889
resource
client-specific
using mwm command 948
resource description file
using mwm command 955
resource set
Client-Specific 949
Component Appearance 932
Frame and Icon Component 934
General Appearance and Behavior 937
run-time linking
using ld command 206

S

screens
displaying text one screen at a time 878
scripts
event response resource manager (ERRM)
logevent 248
logevent 248
session
initiating a user
using login command 253
shared objects
using ld command 191
shells
Korn 173
standard 173
slide
typesetting
using mvt command 924
spool queue
examining
using lpq command 319
spooling queue
removing jobs from the line printer 327
SRC
adding a definition to subsystem object class
mkssys command examples 827
adding a subserver object 823
mkserver command example 824
adding a subsystem notification method 761
mknotify command examples 761
SRC configuration commands
mknotify 761
mkserver 823
mkssys 825
standard input
reading 224
standard output
writing 224
storing public and private keys 161
subserver
getting status
using lssrc command 542
subservers
adding a definition to SRC object class 823
mkserver command example 824
subsystem
getting status
using lssrc command 542
subsystems
adding a definition to SRC object class
mkssys command examples 827
adding a notification method 761
mknotify command examples 761

system
adding a device to
using mkdev command 701
system resource controller 761, 823, 825

T

TCP/IP
hosts
generating table file 723
setting required values 835
internet packets 129
name service
displaying information 466
name services
configuring 749
network interface
interface, configuring 11
network options
displaying, adding, and deleting 871
packet trace report
generating 126
print services
displaying information about 480
service management 43
TCP/IP commands
ifconfig 11
ipreport 126
lsnamsv 466
lsprtsv 480
mkhosts 723
mknamsv 749
mkprtsv 775
mktcpip 835
mmtu 871
TCP/IP daemons
inetd 43
iptrace 129
TCP/IP smit commands
lsnamsv 466
lsprtsv 480
mknamsv 749
mkprtsv 775
mktcpip 835
telinit command 48
terminal
reserving 245
time management
creating reminders 216
TTY
reporting statistics 111
typesetting document
using mmt command 869

U

Unicode 584
updating
procedure
saving files changed during 84
user
initiating a session for a
using login command 253
user database
creating a version for searches
using mkpasswd command 762
users
displaying
system ID 8
displaying attributes
using lsuser command 550
displaying role attributes
using lsrole command 497

V

view graph
typesetting
using mvt command 924
virtual printer
displaying the attribute values of
using lsvirprt command 558
making
using mkvirprt 851
virtual RAM disk
making 783
volume group
displaying information about
using lsvg command 554
displaying information about a physical volume
using lspv command 482
importing new definition 32

W

window
default components 927
default menu
using mwm command 927
iconifying
using mwm command 927
menu pane
using mwm command 963
window manager
event specification
using mwm command 961
Workload Manager (WLM)
listing configurations 569
lswlmconf command 569

X

x defaults 930
XCOFF
modify 212

Printed in USA