Reliable Scalable Cluster Technology for AIX

Technical Reference

SA22-7890-14

Reliable Scalable Cluster Technology for AIX

Technical Reference

SA22-7890-14

Note
Before using this information and the product it supports, read the information in “Notices” on page 619.

Fifteenth Edition (November 2007)

This edition applies to:

- version 5, release 2 of IBM AIX 5L for POWER (product number 5765-E62) with the 5200-10 Technology Level
- version 5, release 3 of IBM AIX 5L for POWER (product number 5765-G03) with the 5300-07 Technology Level
- version 6, release 1 of IBM AIX for POWER (product number 5765-G62) with the 6100-00 Technology Level

and to all subsequent releases and modifications, until otherwise indicated in new editions. Vertical lines (|) in the left margin indicate technical changes to the previous edition of this information.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

IBM welcomes your comments. A form for your comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, Department H6MA, Mail Station P181
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States and Canada): 1+845+432-9405
FAX (Other Countries)
Your International Access Code +1+845+432-9405

IBMLink™ (United States customers only): IBMUSM10(MHVRCFS)
Internet: mhvrcfs@us.ibm.com

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2002, 2007. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this book ix
Who should use this book ix
Conventions and terminology used in this book ix
Conventions ix
Terminology x
Prerequisite and related information x
Using LookAt to find message explanations xii
How to send your comments xii

What's new in RSCT? xiii

Part 1. The resource monitoring and control (RMC) subsystem 1

Chapter 1. RMC control commands 3
chrmcacl 4
rmcctrl 8

Chapter 2. RMC commands 13
chrsrc 14
lsactdef 19
lsrsrc 24
lsrsrcdef 31
mkrsrc 38
refrsrc 43
resetrsrc 45
rmrsrc 50
runact 54
startrsrc 59
stoprsrc 64

Chapter 3. RMC man pages 69
resource_data_input 70
rmccli 74

Part 2. RSCT peer domains: creating and administering 81

Chapter 4. Configuration resource manager commands 83
addrpnode 84
chcomg 88
forcerpoffline 92
lscomg 94
lsrpdomain 98
lsrpnode 101
mkcomg 106
mkrpdomain 110
preprpnode 117
rmcomg 120
rmrpdomain 123
rmrpnode 126
startrpdomain 129
startrpnode 133
stoprpdomain 136

© Copyright IBM Corp. 2002, 2007

stoprpnode 139

Part 3. Cluster configuration 143

Chapter 5. Configuration commands 145
ctadmingroup 146

Chapter 6. Configuration files 149
ctcas_hba2.map 150
ct_class_ids 152
ct_cssk.kf 153
ct_has.pkf 154
ct_has.qkf 156
ct_has.thl 158
ctcasd.cfg 160
ctrmc.acls 165
ctsec.cfg 166
ctsec_map.global, ctsec_map.local 167
unix.map 172

Chapter 7. Common Information Model (CIM) resource manager commands 175
lsassocmap 176
lsrsrcassoc 178
mkcimreg 181

Part 4. Monitoring resources 185

Chapter 8. Event-response resource manager (ERRM) commands 187
chcondition 188
chresponse 193
lscondition 198
lscondresp 204
lsevent 210
lsresponse 215
mkcondition 221
mkcondresp 228
mkresponse 231
rmcondition 237
rmcondresp 240
rmresponse 244
startcondresp 247
stopcondresp 251

Chapter 9. ERRM scripts 255
elogevent, logevent 256
enotifyevent, notifyevent 258
ewallevent, wallevent 260
snmpevent 262

Chapter 10. Sensor resource manager commands 265
chsensor 266
lssensor 270
mksensor 276
refsensor 282

rmsensor 286

Part 5. Auditing resources 289

Chapter 11. Audit log resource manager commands 291
lsaudrec 292
rmaudrec 298

Part 6. Cluster security 303

Chapter 12. Cluster security services commands 305
ctaclfck 306
ctcasd 309
ctmsskf 311
ctscachgen 315
ctscfg 318
ctsidmck 322
ctskeygen 326
ctsthl 329
ctstrtcasd 333
ctsvhbac 335
ctsvhbal 339
ctsvhbar 342

Part 7. Root command management 347

Chapter 13. Least-privilege (LP) resource manager commands 349
chlpcmd 350
lphistory 354
lslpcmd 359
mklpcmd 364
rmlpcmd 369
runlpcmd 372

Chapter 14. LP access control list (ACL) commands 377
chlpclacl 378
chlpracl 383
chlpriacl 389
chlprsacl 394
lslpclacl 399
lslpracl 404
lslpriacl 410
lslprsacl 415

Chapter 15. LP man pages 421
lpacl 422

Part 8. Topology services and group services 431

Chapter 16. Commands for subsystem control and status 433
cthactrl 434
nlssrc 436

Chapter 17. Topology services commands 439
cthatsctrl 440

cthatstune 443
hatsoptions 446
topsvcs 448
topsvcsctrl 449

Chapter 18. Group services commands 453
cthagsctrl 454
cthagstune 458
grpsvcsctrl 460
hagsd 464
hagsns 465
hagsvote 467

Part 9. Virtual shared disk 471

Chapter 19. Virtual shared disk commands 473
cfgvsd 474
createvsd 476
ctlvsd 482
defvsd 485
fencevsd 487
ha_vsd 488
ha.vsd 489
lsvsd 493
preparevsd 496
removevsd 498
resumevsd 500
rvsdrestrict 502
startvsd 504
statvsd 506
stopvsd 508
suspendvsd 510
ucfgvsd 512
undefvsd 514
unfencevsd 515
updatevsdnode 517
updatevsdtab 520
updatevsdvg 521
vsdatalst 524
vsdchgserver 526
vsdelnode 528
vsdelvg 530
vsdnode 532
vsdsklst 535
vsdvg 537
vsdvgts 539

Part 10. Problem determination 541

Chapter 20. Commands for data collection 543
ctsnap 544

Chapter 21. First failure data capture (FFDC) commands 547
fccheck 548
fcclear 550

fcdecode 554
fcdispfid 556
fcfilter 558
fcinit 560
fclogerr 565
fcpushstk 574
fcreport 582
fcstkrpt 585
fcteststk 588

Chapter 22. FFDC files 591
ct_ffdc.h 592

Part 11. The event management subsystem 595

Chapter 23. Event management commands 597
emsvcsctrl 598
haemd 602
haemd_HACMP 603
haemqvar 604
haemtrcoff 608
haemtrcon 610
haemunlkrm 612

Part 12. Appendixes 613

Appendix A. Accessibility features for RSCT 615
Accessibility features 615
Related accessibility information 615
IBM and accessibility 615

Appendix B. Product-related information 617
RSCT version 617
Product-related feedback 617

Notices 619
Trademarks 620

Glossary 623

Index 627

About this book

This book describes Reliable Scalable Cluster Technology (RSCT) commands, daemons, files, and scripts for AIX® 5.2, 5.3, and 6.1.

Who should use this book

This book is intended for system administrators who want to use RSCT commands and files with AIX 5.2, 5.3, or 6.1. The system administrator should be experienced with UNIX® and networked systems.

Conventions and terminology used in this book

Conventions

Table 1 describes the typographic conventions used in this book.

Table 1. Typographic conventions

Convention	Usage
bold	**Bold** words or characters represent system elements that you must use literally, such as: command names, file names, flag names, and path names.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface.
italic	*Italicized* words or characters represent variable values that you must supply. *Italics* are also used for book titles, for the first use of a glossary term, and for general emphasis in text.
{ *item* **}**	Braces indicate required items.
[*item* **]**	Brackets indicate optional items.
*item***...**	Ellipses indicate items that can be repeated.
|	1. In the left margin of the book, vertical lines indicate technical changes to the information. 2. In syntax statements, vertical lines are used as *pipe* characters. See “How to read syntax statements” for more information.

****	In command examples, a backslash indicates that the command continues on the next line. For example:
	
	`mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \`
	`-E "PercentTotUsed < 85" -m d "FileSystem space used"`

How to read syntax statements

Syntax statements represent what you enter on the command line. These statements also contain characters that you do not enter on the command line, which are included to show the various flags and parameters you can enter as part of the command. Such characters include braces (**{ }**), brackets (**[ ]**), and pipe characters (|). For example:

mkcondition -c *existing_condition*[**:***node_name*] [**-r** *resource_class*]
[**-e "***event_expression***"**]
[**-E "***rearm_expression***"**]
[**-d "***event_description***"**]

© Copyright IBM Corp. 2002, 2007

```
[ -D "rearm_description" ]
[-m l | m | p ]
[-n node_name1[,node_name2...]] [-p node_name]
[ --qnotoggle | --qtoggle ]
[-s "selection_string"]  [ -S c | w | i ] [-h] [-TV] condition
```

Syntax statements in this book and in the RSCT man pages use the following conventions:

- Items that must be entered literally on the command line are shown in **bold**. These items include the command name, flags, and other literal characters, such as commas (**,**).
- Variables that you must replace with a value are shown in *italics*. These items include command parameters and parameters that follow flags. For example, the expression **-n** *node_name* might become **-n node1** on the command line.
- Flags and parameters that are enclosed in neither brackets nor braces are required.

  Flags and parameters that are enclosed in braces are also required.
- Optional flags and parameters are enclosed in brackets.
- The pipe character indicates that you choose among the flags shown. For example, the expression { **-a** | **-b** | **-c** } indicates that you *must* choose either **-a**, or **-b**, or **-c**. The expression [**-a** | **-b** | **-c**] indicates that you *can* choose either **-a**, or **-b**, or **-c**.
- Ellipses (**...**) indicate that you can repeat the parameter. For example, the expression:

  [**–n** *node_name1*[**,***node_name2*...]]

  indicates that if you use the **–n** flag, you *must* follow it with at least one node name. You *can* enter as many node names as you'd like, separated by commas. For example:

  ```
  -n node1,node2,node3,node4,node5
  ```

Terminology

This book uses the terminology conventions shown in Table 2:

Table 2. Terminology

Term	Usage
HPS	A shorthand notation for the *High Performance Switch,* which works in conjunction with a specific family of IBM® System p™ servers

See the "Glossary" on page 623 for definitions of some of the other terms that are used in this book.

Prerequisite and related information

The Reliable Scalable Cluster Technology (RSCT) library includes the following publications:

- *RSCT: Administration Guide*, SA22-7889, provides an overview of the RSCT components and describes how to:
  - Create and administer RSCT peer domains.
  - Manage and monitor resources using the resource monitoring and control (RMC) subsystem.

  - Administer cluster security services for RSCT peer domains and CSM management domains.
- *RSCT: Diagnosis Guide*, SA23-2202, describes how to diagnose and resolve problems related to the various components of RSCT. This book is a companion volume to *RSCT: Messages*, which lists the error messages that may be generated by each RSCT component. While *RSCT: Messages* describes the appropriate user responses to messages that are generated by RSCT components, this book contains additional and more detailed diagnostic procedures.
- *RSCT: Group Services Programming Guide*, SA22-7888, contains information for programmers who want to write new clients that use the group services subsystem's application programming interface (GSAPI) or who want to add the use of group services to existing programs. This book is intended for programmers of system management applications who want to use group services to make their applications highly available.
- *RSCT: LAPI Programming Guide*, SA22-7936, provides conceptual, procedural, and reference information about the low-level application programming interface (LAPI). LAPI, a message-passing API that is based on an ″active message style″ mechanism, provides a one-sided communication model and optimal performance across various high-performance switches on AIX and Linux systems.
- *RSCT: NRT API Programming Guide*, SC23-6655, contains information for programmers who want to use the network resource table (NRT) API to work with adapter window resources on non-HPS systems.
- *RSCT for AIX: Managing Shared Disks*, SA22-7937, describes the shared disk management facilities of IBM eServer™ Cluster 1600 server processors — the optional virtual shared disk and recoverable virtual shared disk components of RSCT for AIX. This book describes how you can use these components to manage cluster disks to enable multiple nodes to share the information they hold. The book includes an overview of the components and explains how to plan for them, install them, and use them to add reliability and availability to your data storage.
- *RSCT: Messages*, GA22-7891, lists the error messages that may be generated by each RSCT component. For each message, this manual provides an explanation of the message, and describes how you should respond to it.
- *RSCT: RMC Programming Guide*, SA23-1346, describes the resource monitoring and control application programming interface (RMC API). This book is intended for programmers who want to create applications that use the RMC API to connect to the RMC subsystem to leverage its resource management and monitoring capabilities.
- *RSCT for AIX: Technical Reference*, SA22-7890, and *RSCT for Multiplatforms: Technical Reference*, SA22-7893, provide detailed reference information about all of the RSCT commands, daemons, files, and scripts.

For access to all of the RSCT documentation, refer to the *IBM Cluster information center*:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp

This Web site contains the most recent RSCT documentation in HTML and PDF formats. The *Cluster information center* also includes an *RSCT Documentation Updates* file, which contains documentation corrections and clarifications, as well as information that was discovered after the RSCT books were published. Check this file for pertinent information (about required software patches, for example).

The current RSCT books and earlier versions of the library are also available in PDF format from the *IBM Publications Center* Web site:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp

It is easiest to locate a manual in the *IBM Publications Center* by supplying the manual's publication number. The publication number for each of the RSCT books is listed after the book title in the preceding list.

Using LookAt to find message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM messages you encounter, as well as for some system abends and codes. You can use LookAt from the following locations to find IBM message explanations:

- The Internet. You can access IBM message explanations directly from the LookAt Web site:

  http://www.ibm.com/eserver/zseries/zos/bkserv/lookat
- Your wireless handheld device. You can use the LookAt Mobile Edition with a handheld device that has wireless access and an Internet browser (for example: Internet Explorer for Pocket PCs, Blazer, Eudora for Palm OS, or Opera for Linux® handheld devices). Link to the LookAt Mobile Edition from the LookAt Web site.

How to send your comments

Your feedback is important in helping to provide accurate, high-quality information. If you have any comments about this book or any other RSCT documentation:

- Go to the *IBM Cluster information center* home page at:

  http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp

  Click on the **Contact us** link to go to our feedback page, where you can enter and submit your comments.
- Send your comments by e-mail to: **mhvrcfs@us.ibm.com**

  Include the book title and order number, and, if applicable, the specific location of the information about which you have comments (for example, a page number, table number, or figure number).
- Fill out one of the forms at the back of this book and return it by mail, by fax, or by giving it to an IBM representative.

What's new in RSCT?

Major changes and additions to RSCT include:

- Support for AIX 6.1.
- Support for Windows.

  RSCT 2.5.0 supports IBM Tivoli System Automation for Multiplatforms on the Microsoft Windows platform.
- Support for the VAC 8.0.0.12 compiler.
- A new release (version 4.2) of the virtual shared disk and recoverable virtual shared disk components of RSCT for AIX.
- Group services enhancements:

  You can now define *preferred nodes* for group services nameserver and group leader selection. Use the **mkrpdomain -f** command with a node definition file to specify whether nodes are to be preferred nodes or non-preferred nodes.
- Quorum enhancements:

  You can now define *quorum nodes* to cause quorum calculations to be determined by a subset of nodes called the *quorum set*. Use the **mkrpdomain -f** command with a node definition file to specify whether nodes are to be quorum nodes or non-quorum nodes.
- To minimize the possibility that RSCT daemons are prevented from accessing system resources, the topology services, group services, and configuration resource manager daemons now run with a fixed realtime CPU priority.
- Changed commands: **addrpnode**, **chsensor**, **ctsnap**, **lsrpnode**, **lssensor**, **mkrpdomain**, **mkrsrc**, **mksensor**, **refsensor**, **rmsensor**.

  For RSCT for AIX: **rvsdrestrict**.
- New messages: 2523-897, 2523-898. 2602-028, 2602-029, 2602-030. 2610-234, 2610-444. 2618-011, 2618-047, 2618-086, 2618-087. 2632-148 to 2632-161.
- Changed messages: 2632-054, 2632-072.

© Copyright IBM Corp. 2002, 2007

Part 1. The resource monitoring and control (RMC) subsystem

Chapter 1. RMC control commands 3
chrmcacl 4
rmcctrl 8

Chapter 2. RMC commands 13
chrsrc 14
lsactdef 19
lsrsrc 24
lsrsrcdef 31
mkrsrc 38
refrsrc 43
resetrsrc 45
rmrsrc 50
runact 54
startrsrc 59
stoprsrc 64

Chapter 3. RMC man pages 69
resource_data_input 70
rmccli 74

© Copyright IBM Corp. 2002, 2007

Chapter 1. RMC control commands

© Copyright IBM Corp. 2002, 2007

chrmcacl

Purpose

Updates the resource monitoring and control (RMC) ACL file.

Syntax

chrmcacl [**-a** | **-d** | **-r** | **-h**]

Description

Use this command to update the RMC ACL file (**/var/ct/cfg/ctrmc.acls**). If this file does not exist, **chrmcacl** copies the default ACL file from **/usr/sbin/rsct/cfg/ctrmc.acls** to **/var/ct/cfg/ctrmc.acls**. This command reads update information from standard input. This input must be in ACL file format, so it must consist of one or more stanzas, in which each stanza begins with a stanza name that is followed by zero or more stanza lines. A stanza is terminated by a blank line, a comment line, another stanza, or end-of-file. See the description of the RMC ACL file in the *RSCT: Administration Guide* for details.

With no flags specified, **chrmcacl** does whole stanza addition, replacement, or deletion. If the input stanza does not exist in the ACL file, it is added. If the input stanza has a match in the ACL file, the input stanza replaces the existing ACL file stanza. If the input stanza contains no stanza lines and has a match in the ACL file, the existing ACL file stanza is removed.

If the **-a**, **-r**, or **-d** flag is specified, **chrmcacl** does individual stanza line addition, replacement, or deletion. Stanza lines are matched based on the user identifier and object type tokens, in the stanza line, within matching stanzas. Matches must be exact; in other words, there is no wildcard matching.

When the **-a** flag is used, the permissions specified in the input stanza line are added to the permissions from the matching stanza line in the ACL file. If this results in an effective change in permissions, the new permissions are updated in the ACL file. If there is no matching stanza line in the ACL file, the input stanza line is added to the matching stanza in the ACL file.

When the **-r** flag is used, the input stanza line unconditionally replaces the matching stanza line in the ACL file. If there is no matching stanza line in the ACL file, the input stanza line is added to the matching stanza in the ACL file. For the **-a** and **-r** flags, if the input stanza has no match in the ACL file, the complete input stanza is added to the ACL file.

When the **-d** flag is used, any matching stanza lines in the ACL file are deleted. If, as a result, the matching stanza in the ACL file has no stanza lines, the stanza is removed from the ACL file.

As a by-product of this command, the stanza lines within each stanza are ordered from the most specific user identifiers and object types to less specific user identifiers and object types.

The **chrmcacl** command employs file locking, which is used by other RSCT components, to serialize updates and prevent file corruption. Therefore, it is recommended that you use this command to update the ACL file, rather than by modifying the file directly.

When the ACL file is updated, the previous version is first saved as **/var/ct/cfg/ctrmc.acls.orig**. If there are no effective changes or if there are any errors, the ACL file is not updated.

Changes to the ACL file take effect the next time the RMC subsystem is started. To get the ACL file changes to take effect immediately, run this command:

```
refresh -s ctrmc
```

Flags

–a Adds the permissions of the input stanza lines to the matching stanza lines within the matching ACL file stanzas.

–d Deletes the matching stanza lines within the matching ACL file stanzas.

–r Replaces the matching stanza lines within the matching ACL file stanzas with the input stanza lines.

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

Files

/usr/sbin/rsct/cfg/ctrmc.acls	Default location of the **ctrmc.acls** file
/var/ct/cfg/ctrmc.acls	Location of the modifiable **ctrmc.acls** file
/var/ct/cfg/ctrmc.acls.orig	Location of the previous version of the modifiable **ctrmc.acls** file

Standard input

This command reads update information from standard input.

Standard error

Error messages are written to standard error.

When the **-h** flag is specified, this command's usage statement is written to standard error.

Exit status

0 The command has run successfully.

1 The command was not successful.

Security

Privilege control: only the root user should have execute (**x**) access to this command.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/install/bin/chrmcacl

Examples

1. If the **/var/ct/cfg/ctrmc.acls** file already contains the **IBM.Sensor** stanza, but not the **OTHER** stanza, and given the following input to **chrmcacl** (with no flags specified):

```
IBM.Sensor
    joe@Host1.CoX.com   *    rw
    Host1.CoX.com       *    r

OTHER
    Host1.CoX.com       C    r
```

   the **IBM.Sensor** stanza is replaced by the input stanza and the **OTHER** stanza is added to the file upon successful completion of the command.

2. With the **/var/ct/cfg/ctrmc.acls** file that is a result of example 1 and given the following input to **chrmcacl** (with no flags specified):

```
IBM.Sensor

OTHER
    Host1.CoX.com       *    r
```

   the **IBM.Sensor** stanza is deleted and the **OTHER** stanza is replaced by the input stanza upon successful completion of the command.

3. With the **/var/ct/cfg/ctrmc.acls** file that is a result of example 2 and given the following input to **chrmcacl** (with the **-a** flag specified):

```
OTHER
    Host1.CoX.com       *    w
```

   the **OTHER** stanza in the file will be:

```
OTHER
    Host1.CoX.com       *    rw
```

   upon successful completion of the command.

4. With the **/var/ct/cfg/ctrmc.acls** file that is a result of example 3 and given the same input to **chrmcacl** as in example 3 (with the **-r** flag specified), the **OTHER** stanza in the file will be:

```
OTHER
    Host1.CoX.com       *    w
```

   upon successful completion of the command.

5. Given the following stanza in the **/var/ct/cfg/ctrmc.acls** file:

```
IBM.Sensor
    joe@Host1.CoX.com   C    rw
    joe@Host1.CoX.com   R    r
    Host1.CoX.com       *    r
```

   and the following input to **chrmcacl** (with the **-d** flag specified):

```
IBM.Sensor
    joe@Host1.CoX.com   R    r
```

   the **IBM.Sensor** stanza in the file will be:

```
IBM.Sensor
    joe@Host1.CoX.com   C       rw
        Host1.CoX.com   *       r
```

   upon successful completion of the command.

Related information

Books: *RSCT: Administration Guide*, for information about the RMC ACL file

Files: **ctrmc.acls**

rmcctrl

Purpose

Manages the resource monitoring and control (RMC) subsystem.

Syntax

rmcctrl { **-a** | **-A** | **-d** | **-k** | **-K** | **-m** {**R** | **E** | **D**} | **-M** {**R** | **E** | **D**} | **-p** | **-P** | **-q** | **-Q** | **-s** | **-t** *n* | **-T** | **-u** *n* | **-U** | **-v** *n* | **-V** | **-w** n | **-W** | **-x** | **-X** | **-z** | **-h** }

Description

The **rmcctrl** command controls the operation of the resource monitoring and control (RMC) subsystem. The subsystem is under the control of the system resource controller (SRC) with a subsystem name of **ctrmc** and a subsystem group name of **rsct**. The RMC subsystem definition is added to the subsystem object class and then started when Reliable Scalable Cluster Technology (RSCT) is installed. In addition, an entry is made in the **/etc/inittab** file so that the RMC subsystem is started automatically when the system is booted up.

Note: While the RMC subsystem can be stopped and started by using the **stopsrc** and **startsrc** commands, it is recommended that the **rmcctrl** command be used to perform these functions.

Flags

–a Adds the RMC subsystem to the subsystem object class and places an entry at the end of the **/etc/inittab** file.

–A Adds and starts the RMC subsystem.

–d Deletes the RMC subsystem from the subsystem object class and removes the RMC entry from the **/etc/inittab** file.

–k Stops the RMC subsystem.

–K Stops the RMC subsystem and all resource managers.

–m Specifies the RMC subsystem client message policy. This policy applies to messages sent between the RMC subsystem and any command listed in the *RSCT for AIX 5L: Technical Reference*, when the command is run on a different node than the RMC subsystem (in other words, the CT_CONTACT environment variable is set). These messages are sent using TCP/IP.

This flag is supported on RSCT version 2.3.1.0 or later. The ″Enabled″ policy must be used if the commands are from an earlier version of RSCT.

R Indicates that the client message policy is ″Required″. ″Required″ means that the connection remains open only if message authentication can (and will) be used.

E Indicates that the client message policy is ″Enabled″. ″Enabled″ is the default; message authentication is used if both sides of the connection support it.

D Indicates that the client message policy is ″Disabled″. ″Disabled″ means that message authentication is not used.

–M Specifies the RMC subsystem daemon message policy. This policy applies

to messages sent between the RMC subsystem daemons within a management domain cluster. These messages are sent using the User Datagram Protocol (UDP).

This flag is supported on RSCT release 2.3.5.0 or later (for AIX 5.2) or RSCT release 2.4.1.0 or later (for AIX 5.3). When specified, the indicated message policy takes effect the next time the RMC subsystem is started.

R Indicates that the daemon message policy is ″Required″. ″Required″ means that two daemons communicate only if message authentication can (and will) be used.

E Indicates that the daemon message policy is ″Enabled″. ″Enabled″ is the default; message authentication is used if the sending and receiving daemons support it.

D Indicates that the daemon message policy is ″Disabled″. ″Disabled″ means that message authentication is not used. Disabling message authentication may result in the loss of function if all of the nodes in the cluster are not configured the same.

–p Enables remote client connections.

–P Disables remote client connections.

–q Enables remote client connections the next time the RMC subsystem is started.

–Q Disables remote client connections the next time the RMC subsystem is started.

–s Starts the RMC subsystem.

–t *n* Sets the client message timeout value to *n* seconds. Within this amount of time:

- The first message of the start session protocol must arrive after the RMC subsystem accepts a client connection
- Any complete client message must be received by the RMC subsystem, once the beginning of the message has been received

If either of these time limits is exceeded, the client session is closed. The minimum acceptable value is **10**; the maximum is **86400**.

When specified, this value takes effect the next time the RMC subsystem is started.

–T Sets the client message timeout value to the default value of **10** seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

–u *n* Sets the start session timeout value to *n* seconds. Within this amount of time, the start session processing must complete for a new client session; otherwise, the session is closed. The minimum acceptable value is **60**; the maximum is **86400**.

When specified, this value takes effect the next time the RMC subsystem is started.

–U Sets the start session timeout value to the default value of **300** seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

–v *n* Sets the first command timeout value to *n* seconds. If a first command timer is set when a client session is established with the RMC subsystem, the first command must arrive within the specified number of seconds after the start session processing completes; otherwise, the session is closed. The minimum acceptable value is **10**; the maximum is **86400**.

When specified, this value takes effect the next time the RMC subsystem is started.

–V Sets the first command timeout value to the default value of **10** seconds.

When specified, this value takes effect the next time the RMC subsystem is started.

–w *n* Sets the first command threshold value to *n* client sessions. Once the number of client sessions exceeds this value, the RMC subsystem enables a first command timer on each new, unauthenticated session. If the threshold is set to **0**, the first command timeout function is disabled. The maximum value is **150**.

When specified, this value takes effect the next time the RMC subsystem is started.

–W Sets the first command threshold value to the default value of **150** client sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

–x Enables first command timeouts for non-**root** authenticated client sessions and for unauthenticated client sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

–X Disables first command timeouts for non-root authenticated sessions.

When specified, this value takes effect the next time the RMC subsystem is started.

–z Stops the RMC subsystem and all resource managers, but the command does not return until the RMC subsystem and the resource managers are actually stopped.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Exit status

0 The command has run successfully.

1 The command was not successful.

Security

Privilege control: only the root user should have execute (**x**) access to this command.

Location

/usr/sbin/rsct/bin/rmcctrl

Examples

1. To add the RMC subsystem, enter:
   ```
   rmcctrl -a
   ```
2. To start the RMC subsystem, enter:
   ```
   rmcctrl -s
   ```
3. To stop the RMC subsystem, enter:
   ```
   rmcctrl -k
   ```
4. To delete the RMC subsystem, enter:
   ```
   rmcctrl -d
   ```

Chapter 2. RMC commands

© Copyright IBM Corp. 2002, 2007

chrsrc

Purpose

Changes the persistent attribute values of a resource or a resource class.

Syntax

To change the persistent attribute values of a *resource,* using data that is...

- entered on the command line:

  chrsrc –s *"selection_string"* [**–a** | **–N** { *node_file* | *"***-***"* }] [**–v**] [**–h**] [**–TV**] *resource_class attr***=***value*...

  chrsrc –r [**–v**] [**–h**] [**–TV**] *resource_handle attr***=***value*...

- predefined in an input file:

  chrsrc –f *resource_data_input_file* **–s** *"selection_string"* [**–a** | **–N** { *node_file* | *"***-***"* }] [**–v**] [**–h**] [**–TV**] *resource_class*

  chrsrc –f *resource_data_input_file* **–r** [**–v**] [**–h**] [**–TV**] *resource_handle*

To change the persistent attribute values of a *resource class,* using data that is...

- entered on the command line:

  chrsrc { **–c** | **–C** *domain_name...* } [**–v** [**–a**] [**–h**] [**–TV**] *resource_class attr***=***value*...

- predefined in an input file:

  chrsrc –f *resource_data_input_file* { **–c** | **–C** *domain_name...* } [**–v**] [**–a**] [**–h**] [**–TV**] *resource_class*

Description

The **chrsrc** command changes the persistent attribute values of a resource or a resource class. By default, this command changes the persistent attribute values of a *resource.* Use the **-r** flag to change only the persistent attribute values of the resource that is linked with *resource_handle*. Use the **-s** flag to change the persistent attribute values of all of the resources that match *selection_string*. To change the persistent attributes of a *resource class,* use the **-c** flag.

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.

The **chrsrc** command cannot change dynamic attributes, nor can it change persistent attributes that are designated as **read_only**. 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 *not* designated as **read_only**, use the **-v** flag. When the **chrsrc** command is run with the **-v** flag, the specified attributes are not changed, but are instead merely verified to be persistent and not designated as **read_only**. Once you run **chrsrc -v** to verify that the attributes that are specified on the command line or in *resource_data_input_file* are valid, you can issue the **chrsrc** command without the **-v** flag to actually change the attribute values. Note, however, that just because an attribute *"*passes*"* when **chrsrc -v** is run does not ensure that the attribute can be changed. The underlying resource manager that controls the specified resource determines which attributes can be changed by the **chrsrc** command. After **chrsrc** is run without the **-v** flag, an error message will indicate whether any specified attribute could not be changed.

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

attr=*value*...
: Specifies one or more pairs of attributes and their associated values. *attr* is any defined persistent attribute name. Use the **lsrsrcdef** command to display a list of the defined persistent attributes and their datatypes for the specified resource. The value specified must be the appropriate datatype for the associated attribute. For example, if **NodeNumber** is defined as a **Uint32** datatype, enter a positive numeric value.

  Do not specify this parameter if you run **chrsrc** with the **-f** flag.

resource_class
: Specifies a resource class name. Use the **lsrsrcdef** command to display a list of defined resource class names.

resource_handle
: Specifies a resource handle that is linked with the resource that you want to change. Use the **lsrsrc** command to display a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

Flags

–a
: Specifies that this command applies to 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**.

–c
: Changes the persistent attribute values for *resource_class*.

–C *domain_name...*
: Changes 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 change 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**.

–f *resource_data_input_file*
: Specifies the name of the file that contains resource attribute information.

–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**.

–r Changes the persistent attribute values for the specific resource that matches *resource_handle*.

–s *"selection_string"*
Changes the persistent attribute values for all of the resources that match *selection_string*. *selection_string* must be enclosed within either double or single quotation marks. If *selection_string* contains double quotation marks, enclose it in single quotation marks, for example:

```
-s 'Name == "testing"'
-s 'Name ?= "test"'
```

Only persistent attributes can be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

–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 *not* designated as **read_only**. The **chrsrc** command does *not* change any persistent attribute values 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.

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.

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.

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 No resources were found that match the selection string.

Security

The user needs write permission for the *resource_class* specified in **chrsrc** to run **chrsrc**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chrsrc

Examples

1. To change the **Int32**, **Uint32** and **SD** persistent resource attributes in resource class **IBM.Foo** for the resources that have a **Name** equal to **c175n05**, enter:

```
chrsrc -s 'Name == "c175n05"' IBM.Foo \
Int32=-9999 Uint32=9999\
SD='["testing 1 2 3",1,{2,4,6}]'
```

2. To change the **Int32**, **Uint32** and **SD** resource attributes in resource class **IBM.Foo** for the resource that has a **Name** starting with **c175n**, using a resource data input file with the following contents:

```
PersistentResourceAttributes::
resource 1:
     Int32  = -9999
     Uint32 = 9999
     SD     = ["testing 1 2 3",1,{2,4,6}]
```

   enter:

```
chrsrc -f /tmp/IBM.Foo.chrsrc \
-s 'Name ?= "c175n"' IBM.Foo
```

3. To change the **Name** persistent resource attribute for the resource that has a resource handle equal to ″0x0001 0x4005 0x35ae868c 0x00000000 0xfeef2948 0x0d80b827″, enter:

```
chrsrc -r "0x0001 0x4005 0x35ae868c 0x00000000 0xfeef2948 0x0d80b827" \
Name="c175n05"
```

4. To change the **Int32**, **Uint32** and **SD** persistent resource attributes in resource class **IBM.Foo** for the resources that have a **Name** equal to **Test_Name** on nodes **node1.linwood.com** and **node2.linwood.com** in the cluster, using the **/u/joe/common_nodes** file:

```
# common node file
#
node1.linwood.com      main node
node2.linwood.com      backup node
#
```

   as input, enter:

```
chrsrc -s 'Name == "Test_Name"' -N /u/joe/common_nodes IBM.Foo \
Int32=-9999 Uint32=9999 \
SD='["testing 1 2 3",1,{2,4,6}]'
```

Related information

“rmccli ” on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations and about how to use expressions and selection strings

Commands: **lsrsrc**, **lsrsrcdef**, **mkrsrc**, **nodegrp**, **rmrsrc**

lsactdef

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.

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.

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 datatypes 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.

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.

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.

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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsactdef

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
```

Related information

“rmccli ” on page 74, for general information about RMC commands

Books:

- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrcdef**

lsrsrc

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:

- use the **-A p** | **d** | **b** flag to control 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.

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.

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 datatype 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.

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.

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.

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.

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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsrsrc

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 four 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.rnd.unixrulz.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.rnd.unixrulz.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
   ```

Related information

“rmccli ” on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrcdef**, **mkrsrc**, **nodegrp**

lsrsrcdef

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.

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.

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 datatype 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.

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.

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.

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.

Security

The user needs write permission for the *resource_class* specified in **lsrsrcdef** to run **lsrsrcdef**. Permissions are specified in the access control list (ACL) file on the contacted system. see the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsrsrcdef

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:
...
```

Related information

- “resource_data_input ” on page 70
- “rmccli ” on page 74, for general information about RMC commands

Books:

- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **mkrsrc**

mkrsrc

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**.

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 datatype for this value must match the defined datatype for the value of this attribute. Use the **lsrsrcdef** command to verify the datatype for each attribute.

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.

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.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

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.

Security

The user needs write permission for the *resource_class* specified in **mkrsrc** to run **mkrsrc**. Permissions are specified in the access control list (ACL) file on the contacted system. see the *RSCT: Administration Guide* for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mkrsrc

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** is a required attribute. The **Binary**, **SD**, **StringArray**, and **SDArray** attributes are optional. This example shows how to enter the more complex datatypes from the command line. The datatypes for the optional attributes (**Binary**, **SD**, **StringArray**, and **SDArray**) are self-explanatory. Enter:

```
mkrsrc IBM.Foo Name=c175n05 \
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.Foo** class on multiple nodes in a peer domain, run this command:

   ```
   mkrsrc -N /u/joe/common_node_file  IBM.Foo  Name=Foo_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.Foo** class on multiple managed nodes in a management domain, run this command on the management server:

   ```
   mkrsrc IBM.Foo  Name=Foo_bar1 Binary="0xaabbccddeeff00" \
   NodeNameList='{"mgnode1.ibm.com","mgnode2.ibm.com"}'
   ```

Note: As discussed in “rmccli ” on page 74, attribute values for certain datatypes (structured data, arrays of structured data, and arrays containing strings enclosed in double quotation marks) should be enclosed in single quotation marks.

Related information

- “resource_data_input ” on page 70
- “rmccli ” on page 74, for general information about RMC commands

Books:

- *RSCT: Administration Guide*, for information about RMC operations

Commands: **chrsrc**, **lsrsrc**, **lsrsrcdef**, **rmrsrc**

refrsrc

Purpose

Refreshes the resources within the specified resource class.

Syntax

refrsrc [**–h**] [**–TV**] *resource_class*

Description

The **refrsrc** command refreshes the resources within the specified resource class. Use this command to force the resource monitoring and control (RMC) subsystem to detect new instances of resources in cases where the configuration could be altered by operating system commands (**mkfs**, for example).

This command makes a request to the RMC subsystem to refresh the configuration of the resources within a resource class. The request is actually performed by the linked resource manager.

Any application that is monitoring resources in the specified resource class may receive events as the configuration is refreshed.

Parameters

resource_class Specifies the resource class name.

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.

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.

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.

Security

To run the **refrsrc** command, you need read permission for the resource class that you specify with the command. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/refrsrc

Examples

1. To refresh the configuration of the resources in class IBM.FileSystem, enter:

```
refrsrc IBM.FileSystem
```

Related information

“rmccli ” on page 74, for general information about RMC commands

Books:

- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **lsrsrcdef**

resetrsrc

Purpose

Resets a resource (that is, forces it offline).

Syntax

To reset one or more resources, using data entered on the command line:

resetrsrc -s *"selection_string"* [**–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class* [*arg*=*value...*]

resetrsrc -r [**–h**] [**–TV**] *resource_handle* [*arg*=*value...*]

To reset one or more resources using command arguments that are predefined in an input file:

resetrsrc -f *resource_data_input_file* **-s** *"selection_string"* [**–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class*

resetrsrc -f *resource_data_input_file* **-r** [**–h**] [**–TV**] *resource_handle*

To display the names and datatypes of the command arguments:

resetrsrc -l [**–h**] *resource_class*

Description

The **resetrsrc** command requests that the resource monitoring and control (RMC) subsystem force one or more resources offline. The request is actually performed by the appropriate resource manager.

To reset one or more resources, use the **-s** flag to force offline all of the resources that match the specified selection string. To reset one specific resource, use the **-r** flag to specify the resource handle that represents that specific resource.

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.

Use the **-l** flag to determine whether the specified resource class accepts any additional command arguments.

The successful completion of this command does not guarantee that the resource is offline, only that the resource manager successfully received the request to force this resource offline. Monitor the resource's dynamic attribute **OpState** to determine when the resource is actually forced offline. Register an event for the resource, specifying the **OpState** attribute, to know when the resource is actually offline. Or, intermittently run the **lsrsrc** command until you see that the resource is offline (the value of **OpState** is **2**). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

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

resource_class
Specifies the name of the resource class that contains the resources that you want to force offline.

resource_handle
Specifies the resource handle that corresponds to the resource you want to force offline. Use the **lsrsrc** command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=*value...*
Specifies one or more pairs of command argument names and values.

arg Specifies the argument name.

value Specifies the value for this argument. The value's datatype must match the definition of the argument's datatype.

Command arguments are optional. If any *arg*=*value* pairs are entered, there should be one *arg*=*value* pair for each of the command arguments defined for the offline function for the specified resource class.

Use **resetrsrc -l** to get a list of the command argument names and datatypes for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information. The contents of the file would look like this:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the offline 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 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**.

-r Forces offline the specific resource that matches the specified resource handle.

-s *"selection_string"*
Specifies the 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 can be listed in a selection string.

–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.

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.

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 No resources were found that match the specified selection string.

Security

The user needs write permission for the *resource_class* specified in **resetrsrc** to run **resetrsrc**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/resetrsrc

Examples

Suppose you have a peer domain called **foo** with three defined nodes: **nodeA**, **nodeB**, and **nodeC**. **nodeA** has two Ethernet cards: **ent0** and **ent1**.

1. Suppose **nodeA** is online and **ent0** (on **nodeA**) is also online. To force **ent0** offline on **nodeA**, run this command on **nodeA**:

   ```
   resetrsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose **nodeA** and **nodeB** are online, **ent0** (on **nodeA**) is also online, and you are currently logged on to **nodeB**. To force **ent0** offline on **nodeA**, run this command on **nodeB**:

   ```
   resetrsrc -s 'NodeName == "nodeA" AND Name == "ent0"' IBM.EthernetDevice
   ```

3. Suppose **nodeA** and **nodeB** are online and file system **/filesys1** is defined and mounted on **nodeB**. To force **/filesys1** offline on **nodeB**, run this command on **nodeA**:

   ```
   resetrsrc -s 'NodeName == "nodeB" AND Name == "/filesys1"' IBM.FileSystem
   ```

4. Suppose the resource handle for **ent0** on **nodeA** is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To force **ent0** offline on **nodeA**, run this command on **nodeA**:

```
resetrsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
```

5. To reset **ent0** on **nodeA** and **nodeB**, using the **/tmp/common/node_file** file:

   ```
   # common node file
   #
   nodeA
   nodeB
   #
   ```

   as input, enter:

   ```
   resetrsrc -s 'Name == "ent0"' -N /tmp/common/node_file \
   IBM.EthernetDevice
   ```

Related information

“resource_data_input ” on page 70

“rmccli ” on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **nodegrp**, **startrsrc**, **stoprsrc**

rmrsrc

Purpose

Removes a defined resource.

Syntax

To remove one or more resources...

- entered on the command line:

  rmrsrc –s *"selection_string"* [**–a** | **–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class*

  rmrsrc –r *"resource_handle"* [**–h**] [**–TV**]

- predefined in an input file:

  rmrsrc –f *resource_data_input_file* **–s** *"selection_string"* [**–a** | **–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class*

  rmrsrc –f *resource_data_input_file* **–r** *"resource_handle"* [**–h**] [**–TV**]

To display the names and datatypes of the command arguments:

rmrsrc –l [**–h**] *resource_class*

Description

The **rmrsrc** command removes — or *"*undefines*"* — the specified resource instance (or instances). The **rmrsrc** command makes a request to the resource monitoring and control (RMC) subsystem to undefine a specific resource instance. The resource manager of the resource removes the resource.

The first format of this command requires a resource class name parameter and a selection string specified using the **-s** flag. All resources in the specified resource class that match the specified selection string are removed. If the selection string identifies more than one resource to be removed, it is the same as running this command once for each resource that matches the selection string.

The second format of this command allows the actual resource handle linked with a specific resource to be specified as the parameter. It is expected that this form of the command would be more likely used from within a script.

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.

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

resource_class
: Specifies the resource class name. The resource instances for this resource class that match the selection string criteria are removed.

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, **rmrsrc -a** with **CT_MANAGEMENT_SCOPE** not set will apply to the management domain. In this case, to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to **2**.

–f *resource_data_input_file*
Specifies the name of the file that contains resource argument information.

-l Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the remove 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 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**.

–r *"resource_handle"*
Specifies a resource handle. The resource handle must be specified in this format:

*"***0x***nnnn* **0x***nnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn"*

where *n* is a hexadecimal character. The resource handle uniquely identifies a particular resource instance that should be removed.

–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 can be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

–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.

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.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

The command output and all verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

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 No resources were found that match the selection string.

Security

To run the **rmrsrc** command, you need write permission for the resource class that you specify with the command. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmrsrc

Examples

1. To remove the resource named **c175n05** from the **IBM.Host** resource class, enter:

   ```
   rmrsrc -s 'Name == "c175n05"' IBM.Host
   ```

2. To remove the resource that is linked with resource handle **0x4017 0x0001 0x00000000 0x0069684c 0x0d52332b3 0xf3f54b45**, enter:

   ```
   rmrsrc -r "0x4017 0x0001 0x00000000 0x0069684c 0x0d52332b3 0xf3f54b45"
   ```

3. To remove the resources named **Test1** from **IBM.Foo** for certain nodes in the cluster, 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:

   ```
   rmrsrc -s 'Name == "Test1"' -N /tmp/common/node_file IBM.Foo
   ```

Related information

“rmccli ” on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **mkrsrc**, **nodegrp**

runact

Purpose

Runs an action on a resource class.

Syntax

runact –s *"selection_string"* [**–N** { *node_file* | *"-"* }] [**–f** *resource_data_input_file*] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

runact –r [**–f** *resource_data_input_file*] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] *resource_handle action* [*in_element*=*value*...] [*rsp_element*...]

runact –c [**–f** *resource_data_input_file*] [**–n** *node_name*] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

runact –C *domain_name*... [**–f** *resource_data_input_file*] [**–l** | **–t** | **–d** | **–D** *delimiter*] [**–x**] [**–h**] [**–TV**] *resource_class action* [*in_element*=*value*...] [*rsp_element*...]

Description

The **runact** command requests that the RMC subsystem run the specified action on the specified 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.

Before you run this command, use the **lsactdef** command to list the resource class actions that are supported by this resource class. Also, use the **lsactdef** command to list the required input action elements that must be specified when invoking an action. The **lsactdef** command also identifies the datatype for each input element. The value specified for each input element must match this datatype.

Parameters

action Specifies the name of the action to be invoked.

in_element=*value*...
Specifies the action input element names and values. If you use the **-f** flag, don't enter any *in_element*=*value* pairs on the command line.

in_element is any of the input structured data element names. There should be one *in_element_n*=*value* pair for each of the defined structured data (SD) input elements for the specified action. Use **lsactdef** with the **-s i** flag to list the input elements for a particular resource class and action. Use **lsactdef -i** to generate an input file template, which, after appropriate editing, can be used as the input file.

value must be the appropriate datatype for the specified element. For example, if **NodeNumber** is defined as a **uint32** datatype, enter a positive numeric value.

resource_class
Specifies the name of the resource class with the actions that you want to invoke.

resource_handle
Specifies the resource handle for the resource and class with the actions that you want to invoke.

rsp_element Specifies one or more of action response structured data element names. If you specify one or more element names, only those elements are displayed in the order specified. If you do not specify any element names, all elements of the response are displayed.

Flags

–c Invokes the action on the resource class.

To invoke the class action on a globalized resource class on all peer domains defined on the management server, set **CT_MANAGEMENT_SCOPE=3** and use the **-c** flag.

–C *domain_name...*
Invokes a class action on 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.

–f *resource_data_input_file*
Specifies the name of the file that contains resource action input elements and values. Use the **lsactdef** command with the **-i** flag to generate a template for this input file.

–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.

–l Specifies ″long″ format — one entry per line. This is the default display format.

–n *node_name*
Specifies the name of the node on which to run the class action. You can only use this flag in conjunction with the **-c** flag.

–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**.

–r *"resource_handle"*
Specifies a resource handle. The resource handle must be specified in this format:

*"***0x***nnnn* **0x***nnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn"*

where *n* is a hexadecimal character. Use this flag to invoke the action on the resource that matches *resource_handle*.

–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 can 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.

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.

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.

Security

This command requires **root** authority.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/runact

Examples

1. To invoke the TestClassAction resource class action on the resource class **IBM.Foo**, enter:

```
runact -c IBM.Foo TestClassAction Int32=99
```

The output will look like this:

```
Resource Class Action Response for: TestClassAction
sd_element 1:
   Int32 = 99
```

Related information

- “resource_data_input ” on page 70
- “rmccli ” on page 74, for general information about RMC commands

Books:

- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsactdef**

startrsrc

Purpose

Starts a defined resource (that is, brings it online).

Syntax

To start one or more resources, using data entered on the command line:

startrsrc -s *"selection_string"* [**–N** { *node_file* | *"-"* }] [**-n** *node_name*] [**–h**] [**–TV**] *resource_class* [*arg*=*value...*]

startrsrc -r [**-n** *node_name*] [**–h**] [**–TV**] *resource_handle* [*arg*=*value...*]

To start one or more resources using command arguments that are predefined in an input file:

startrsrc -f *resource_data_input_file* **-s** *"selection_string"* [**–N** { *node_file* | *"-"* }] [**-n** *node_name*] [**–h**] [**–TV**] *resource_class*

startrsrc -f *resource_data_input_file* **-r** [**-n** *node_name*] [**–h**] [**–TV**] *resource_handle*

To list the names and datatypes of the command arguments:

startrsrc -l [**–h**] *resource_class*

Description

The **startrsrc** command requests that the resource monitoring and control (RMC) subsystem bring one or more resources online. The request is actually performed by the appropriate resource manager.

To start one or more resources, use the **-s** flag to bring online all of the resources that match the specified selection string.

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.

To start one specific resource, use the **-r** flag to specify the resource handle that represents that specific resource.

Use the **-l** flag to determine whether the specified resource class accepts any additional command arguments.

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 successful completion of this command does not guarantee that the resource is online, only that the resource manager successfully received the request to bring this resource online. Monitor the resource's dynamic attribute **OpState** to determine when the resource is actually brought online. Register an event for the resource,

specifying the **OpState** attribute, to know when the resource is actually online. Or, intermittently run the **lsrsrc** command until you see that the resource is online (the value of **OpState** is **1**). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

Parameters

resource_class
Specifies the name of the resource class that contains the resources that you want to bring online.

resource_handle
Specifies the resource handle that corresponds to the resource you want to bring online. Use the **lsrsrc** command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=*value...*
Specifies one or more pairs of command argument names and values.

arg Specifies the argument name.

value Specifies the value for this argument. The value's datatype must match the definition of the argument's datatype.

Command arguments are optional. If any *arg*=*value* pairs are entered, there should be one *arg*=*value* pair for each of the command arguments defined for the online function for the specified resource class.

Use **startrsrc -l** to get a list of the command argument names and datatypes for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information. The contents of the file would look like this:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the online request. Use this flag to list any defined command arguments and the datatypes of the command argument values.

-n *node_name*
Specifies the node name of the node where the resource is to be brought online. Use this flag to bring a floating resource online on a different node when the node where it had been online may be down. The node name is one of the names contained in the **NodeNameList** attribute.

Do *not* specify this flag if you want the resource to be brought online on the node where it is known.

–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**.

-s *"selection_string"*
Specifies the 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 can be listed in a selection string.

–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.

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.

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 No resources were found that match the specified selection string.

Security

The user needs write permission for the *resource_class* specified in **startrsrc** to run **startrsrc**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/startrsrc

Examples

Suppose you have a peer domain called **foo** with three defined nodes: **nodeA**, **nodeB**, and **nodeC**. **nodeA** has two Ethernet cards: **ent0** and **ent1**.

1. Suppose **nodeA** is online and **ent0** (on **nodeA**) is offline. To bring **ent0** online on **nodeA**, run this command on **nodeA**:

   ```
   startrsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose **nodeA** and **nodeB** are online, **ent0** (on **nodeA**) is offline, and you are currently logged on to **nodeB**. To bring **ent0** online on **nodeA**, run this command on **nodeB**:

   ```
   startrsrc -s 'Name == "ent0'" -n nodeA IBM.EthernetDevice
   ```

3. Suppose file system **/filesys1** is defined, but not mounted on **nodeB**. To bring **/filesys1** online on **nodeB**, run this command on **nodeA**:

   ```
   startrsrc -s 'Name == "/filesys1"' -n nodeB IBM.FileSystem
   ```

4. Suppose the resource handle for **ent0** on **nodeA** is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To bring **ent0** online on **nodeA**, run this command on **nodeA**:

   ```
   startrsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
   ```

Related information

"resource_data_input " on page 70

"rmccli " on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **nodegrp**, **resetrsrc**, **stoprsrc**

stoprsrc

Purpose

Stops a resource (that is, takes it offline).

Syntax

To stop one or more resources, using data entered on the command line:

stoprsrc -s *"selection_string"* [**–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class* [*arg*=*value...*]

stoprsrc -r [**–h**] [**–TV**] *resource_handle* [*arg*=*value...*]

To stop one or more resources using command arguments that are predefined in an input file:

stoprsrc -f *resource_data_input_file* **-s** *"selection_string"* [**–N** { *node_file* | *"-"* }] [**–h**] [**–TV**] *resource_class*

stoprsrc -f *resource_data_input_file* **-r** [**–h**] [**–TV**] *resource_handle*

To list the names and datatypes of the command arguments:

stoprsrc -l [**–h**] *resource_class*

Description

The **stoprsrc** command requests that the resource monitoring and control (RMC) subsystem take one or more resources offline. The request is actually performed by the appropriate resource manager.

To stop one or more resources, use the **-s** flag to take offline all of the resources that match the specified selection string.

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.

To stop one specific resource, use the **-r** flag to specify the resource handle that represents that specific resource.

Use the **-l** flag to determine whether the specified resource class accepts any additional command arguments.

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 successful completion of this command does not guarantee that the resource is offline, only that the resource manager successfully received the request to take this resource offline. Monitor the resource's dynamic attribute **OpState** to determine when the resource is actually taken offline. Register an event for the resource,

specifying the **OpState** attribute, to know when the resource is actually offline. Or, intermittently run the **lsrsrc** command until you see that the resource is offline (the value of **OpState** is **2**). For example:

```
lsrsrc -s 'Name == "/filesys1"' -t IBM.FileSystem Name OpState
```

Parameters

resource_class
Specifies the name of the resource class that contains the resources that you want to take offline.

resource_handle
Specifies the resource handle that corresponds to the resource you want to take offline. Use the **lsrsrc** command to obtain a list of valid resource handles. The resource handle must be enclosed within double quotation marks, for example:

```
"0x4017 0x0001 0x00000000 0x0069684c 0x0d4715b0 0xe9635f69"
```

arg=*value...*
Specifies one or more pairs of command argument names and values.

arg Specifies the argument name.

value Specifies the value for this argument. The value's datatype must match the definition of the argument's datatype.

Command arguments are optional. If any *arg*=*value* pairs are entered, there should be one *arg*=*value* pair for each of the command arguments defined for the offline function for the specified resource class.

Use **stoprsrc -l** to get a list of the command argument names and datatypes for the specific resource class.

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource attribute information. The contents of the file would look like this:

```
PersistentResourceArguments::
argument1 = value1
argument2 = value2
```

-l Lists the command arguments and datatypes. Some resource managers accept additional arguments that are passed to the offline 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 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**.

-s *"selection_string"*
Specifies the 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 can be listed in a selection string.

–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.

Environment variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource conitoring 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.

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 No resources were found that match the specified selection string.

Security

The user needs write permission for the *resource_class* specified in **stoprsrc** to run **stoprsrc**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/stoprsrc

Examples

Suppose you have a peer domain called **foo** with three defined nodes: **nodeA**, **nodeB**, and **nodeC**. **nodeA** has two Ethernet cards: **ent0** and **ent1**.

1. Suppose **nodeA** is online and **ent0** (on **nodeA**) is also online. To take **ent0** offline on **nodeA**, run this command on **nodeA**:

   ```
   stoprsrc -s 'Name == "ent0"' IBM.EthernetDevice
   ```

2. Suppose **nodeA** and **nodeB** are online, **ent0** (on **nodeA**) is also online, and you are currently logged on to **nodeB**. To take **ent0** offline on **nodeA**, run this command on **nodeB**:

   ```
   stoprsrc -s 'NodeName == "A" AND Name == "ent0"' IBM.EthernetDevice
   ```

3. Suppose **nodeA** and **nodeB** are online and file system **/filesys1** is defined and mounted on **nodeB**. To take **/filesys1** offline on **nodeB**, run this command on **nodeA**:

   ```
   stoprsrc -s 'NodeName == "B" AND Name == "/filesys1"' IBM.FileSystem
   ```

4. Suppose the resource handle for **ent0** on **nodeA** is:

   ```
   0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e
   ```

   To take **ent0** offline on **nodeA**, run this command on **nodeA**:

```
stoprsrc -r "0x406b 0x0001 0x00000000 0x0069564c 0x0dc1f272 0xb9de145e"
```

Related information

"resource_data_input " on page 70

"rmccli " on page 74, for general information about RMC commands

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **lsrsrc**, **nodegrp**, **resetrsrc**, **startrsrc**

Chapter 3. RMC man pages

© Copyright IBM Corp. 2002, 2007

resource_data_input

Purpose

Describes how to use an input file for passing resource class information, such as resource attribute names and values, to the resource monitoring and control (RMC) command-line interface (CLI).

Description

You can use the **-f** flag with most RMC commands to specify the name of a resource data input file when you want to pass resource persistent attribute values and other information to the RMC CLI. This is useful when typing information on the command line would be too cumbersome or too prone to typographical errors. The data in this file is used for defining resources or for changing the persistent attribute values of a resource or resource class. This file has no set location. It can be a temporary file or a permanent file, depending on your requirements.

The **chrsrc**, **mkrsrc**, **resetrsrc**, **rmrsrc**, **runact**, **startrsrc**, and **stoprsrc** commands read this file when they are issued with the **-f** flag. The **lsactdef**, **lsrsrc**, and **lsrsrcdef** commands generate a file with this format when they are issued with the **-i** flag.

Keywords are used in the input file to indicate which type of data is listed in the related stanza:

ResourceAction
: Resource action element names and values for the resource action when invoking an action. The **runact** command reads in the resource action elements. These elements are ignored if the input file is read by **runact -c**.

ResourceClassAction
: Resource class action element names and values for the resource class action when invoking an class action. The **runact** command reads in the resource action elements.

PersistentResourceArguments
: Resource command argument names and values for those commands that accept them: **mkrsrc**, **resetrsrc**, **rmrsrc**, **startrsrc**, and **stoprsrc**. Command arguments are optional and are defined by the resource class. Specify the **-l** option with these commands to see the command arguments for a resource class.

PersistentResourceAttributes
: Persistent attribute names and values for one or more resources for a specific resource class used to define a new resource or change attribute values for an existing resource. The persistent resource attributes are read in by the commands **mkrsrc** and **chrsrc**. These attributes are ignored if the input file is read by the **chrsrc** command that has been specified with the **-c** flag.

PersistentResourceClassAttributes
: Persistent attribute names and values for a resource class used to change the attribute values of an existing resource class. The persistent resource class attributes are read in by the command **chrsrc** only when the **-c** flag is specified.

In general, a *resource_data_input* file is a flat text file with the following format. **Bold** words are literal. Text that precedes a single colon (**:**) is an arbitrary label and can be any alphanumeric text.

```
PersistentResourceAttributes::
# This is a comment
    label:
      AttrName1  = value
      AttrName2  = value
      AttrName3  = value
    another label:
        Name        = name
        NodeNumber  = 1

⋮
 ::

PersistentResourceClassAttributes::
# This is a comment
    label:
      SomeSettableAttrName   = value
      SomeOtherSettableAttrName  = value
    ::
⋮

PersistentResourceArguments::
# This is a comment
    label:
      ArgName1  = value
      ArgName2  = value
      ArgName3  = value
    ::
⋮
```

See the **Examples** section for more details.

Some notes about formatting follow:

- The keywords **PersistentResourceAttributes**, **PersistentResourceClassAttributes**, and **PersistentResourceArguments** are followed by two colons (**::**).
- The order of the keyword stanzas is not significant in the file. For example, **PersistentResourceClassAttributes** could precede **PersistentResourceClass**. It does not affect the portion of the data that is read in by the calling CLI.
- Individual stanza headings (beneath the keywords) are followed by one colon (**:**), for example: `c175n05 resource info:`
- White space at the beginning of lines is not significant. Tabs or spaces are suggested for readability.
- Any line with a pound sign (**#**) as the first printable character is a comment.
- Each entry on an individual line is separated by white space (spaces or tabs).
- Blank lines in the file are not significant and are suggested for readability.
- There is no limit to the number of resource attribute stanzas included in a particular **PersistentResourceAttributes** section.
- There is no limit to the number of resource class attribute stanzas included in a particular **PersistentResourceClassAttributes** section. Typically, there is only one instance of a resource class. In this case, only one stanza is expected.
- If only one resource attribute stanza is included in a particular **PersistentResourceAttributes** section, the *label:* line can be omitted. This also applies to the **ResourceAction** section.

- If only one resource class attribute stanza is included in a particular **PersistentResourceClassAttributes** section, the *label:* line can be omitted. This also applies to the **ResourceClassAction** section.
- Values that contain spaces must be enclosed in quotation marks.
- A double colon (**::**) indicates the end of a section. If a terminating double colon is not found, the next **Reserved Keyword** or **end of file** signals the end of a section.
- Double quotation marks included within a string that is surrounded by double quotation marks must be escaped. (\″).

  Note: Double quotation marks can be nested within single quotation marks. These are examples:

  – `"Name == \"testing\""`

  – `'Name == "testing"'`

    This syntax is preferred if your string is a selection string and you are going to cut and paste to the command line.
- Single quotation marks included within a string that is surrounded by single quotation marks must be escaped. (\’).

  Note: Single quotation marks can be nested within double quotation marks. Here are some examples:

  – `'Isn\'t that true'`

  – `"Isn't that true"`

    This syntax is preferred if you are going to cut and paste to the command line.
- The format you use to enter data in a *resource_data_input* file may not be the same format used on the command line. The shell you choose to run the commands in has its own rules with regard to quotation marks. Refer to the documentation for your shell for these rules, which determine how to enter data on the command line.

Implementation specifics

This man page is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/man/resource_data_input.7

Examples

1. This sample **mkrsrc** command:

   ```
   mkrsrc -f /tmp/my_resource_data_input_file IBM.Foo
   ```

   uses the sample input file **/tmp/my_resource_data_input_file** for the **IBM.Foo** resource class. The contents of the input file look like this:

   ```
   PersistentResourceAttributes::
   # Resource 1 - only set required attributes
   resource 1:
       Name="c175n04"
       NodeList = {1}
   # Resource 2 - setting both required and optional attributes
   # mkrsrc -e2 IBM.Foo displays required and optional
   # persistent attributes
   ```

```
resource 2:
    Name="c175n05"
    NodeList = {1}
    Int32 = -99
    Uint32 = 99
    Int64 = -123456789123456789
    Uint64 = 123456789123456789
    Float32 = -9.89
    Float64 = 123456789.123456789
    String = "testing 123"
    Binary = 0xaabbccddeeff
    RH = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
    SD = [hello,1,{2,4,6,8}]
    Int32Array = {-4, -3, -2, -1, 0, 1, 2, 3, 4}
    Int64Array = {-4,-3,-2,-1,0,1,2,3,4}
    Uint32Array = {0,1,2,3,4,5,6}
    Uint64Array = {0,1,2,3,4,5,6}
    Float32Array = {-3.3, -2.2, -1.2, 0, 1, 2.2, 3.3}
    Float64Array = {-3.3, -2.2, -1.2, 0, 1, 2.2, 3.3}
    StringArray = {abc,"do re mi", 123}
    BinaryArray = {"0x01", "0x02", "0x0304"}
    RHArray     = {"0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000",
                   "0xaaaa 0xaaaa 0xbbbbbbbb 0xcccccccc 0xdddddddd 0xeeeeeeee"}
    SDArray     = {[hello,1,{0,1,2,3}],[hello2,2,{2,4,6,8}]}
```

2. This sample **chrsrc** command:

   ```
   chrsrc -f  /tmp/Foo/ch_resources -s 'Name == "c175n05"' IBM.Foo
   ```

   uses the sample input file **/tmp/Foo/ch_resources** to change the attribute values of existing **IBM.Foo** resources. The contents of the input file look like this:

   ```
   PersistentResourceAttributes::
   # Changing resources that match the selection string entered
   # when running chrsrc command.
    resource 1:
           String            = "this is a string test"
           Int32Array        = {10,-20,30,-40,50,-60}
   ```

3. This sample **rmrsrc** command:

   ```
   rmrsrc -l IBM.Foobar
   ```

   shows the optional command arguments:

   ```
   rmrsrc IBM.Foobar FooInt32=int32 FooUint32=uint32
   ```

4. This sample **rmrsrc** command:

   ```
   rmrsrc -f /tmp/Foobar/rm_resources -s 'Name == "c175n05"' IBM.Foobar
   ```

   uses the sample input file **/tmp/Foobar/rm_resources** to specify the optional command arguments for **rmrsrc**. The contents of the input file look like this:

   ```
   PersistentResourceArguments::
   # Specifying command arguments when running rmrsrc command.
   resource 1:
    FooInt32      =   1
           FooUint32     =   0
   ```

Related information

Commands: **chrsrc**, **lsactdef**, **lsrsrcdef**, **mkrsrc**, **rmrsrc**

“rmccli ” on page 74

rmccli

Purpose

Provides general information about resource monitoring and control (RMC) and related commands.

Description

This man page provides general information about RMC and related commands, including datatypes, terminology, and references to related information.

Command structure and use

The RMC commands may be grouped into categories representing the different operations that can be performed on resource classes and resources:

- Creating and removing resources: **mkrsrc**, **rmrsrc**
- Modifying resources: **chrsrc**, **refrsrc**
- Viewing definitions and data: **lsrsrc**, **lsrsrcdef**
- Viewing actions: **lsactdef**
- Running actions: **runact**

The RMC commands can be run directly from the command line or called by user-written scripts. In addition, the RMC commands are used as the basis for higher-level commands, such as the event response resource manager (ERRM) commands.

Data display information

The flags that control the display function for the RMC CLI routines, in order of precedence, are:

1. **–l** for long display. This is the default display format.

   For example, the command:

   ```
   lsrsrc -s 'Name == "c175n05"' IBM.Foo Name NodeList SD Binary RH Int32Array
   ```

   produces output that looks like this:

```
Persistent Attributes for Resource: IBM.Foo
resource 1:
        Name       = "c175n05"
        NodeList   = {1}
        SD         = ["testing 1 2 3",1,{0,1,2}]
        Binary     = "0xaabbcc00 0xeeff"
        RH         = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
        Int32Array = {1,5,-10,1000000}
```

2. **–t** for tabular display.

   For example, the command:

   ```
   lsrsrc -s 'Name ?= "Page"' -t IBM.Condition Name EventExpression
   ```

   produces output that looks like this:

   ```
   Persistent Attributes for Resource: IBM.Condition

   Name                  EventExpression
   "Page space out rate" "VMPgSpOutRate > 500"
   "Page fault rate"     "VMPgFaultRate > 500"
   "Page out rate"       "VMPgOutRate > 500"
   "Page in rate"        "VMPgInRate > 500"
   "Page space in rate"  "VMPgSpInRate > 500"
   ```

3. **–x** for suppressing headers when printing.

4. **–d** for colon (:) delimited display.
   For example, the command:
   ```
   lsrsrc -xd -s 'Name == "c175n05"' IBM.Foo Name Int32 Uint32Array SD Binary
   ```
   produces output that looks like this:
   ```
   c175n05:-100:{}:["hel  lo1",1,{0,1,2}]:"0xaabbcc00 0xeeff":
   ```
   Note the use of the **–x** flag along with the **–d** flag.
5. **–D** *delimiter* for string-delimited display.
   For example, the command:
   ```
   lsrsrc -xD:: -s 'Name == "c175n05"' IBM.Foo Name Int32 Uint32Array SD Binary
   ```
   produces output that looks like this:
   ```
   c175n05::-100::{}::["hel  lo1",1,{0,1,2}]::"0xaabbcc00 0xeeff"::
   ```
   Note the use of the **–x** flag along with the **–D** *Delimiter* flag.

When output of any list command (**lsrsrc**, **lsrsrcdef**) is displayed in the tabular output format, the printing column width may be truncated. If more characters need to be displayed (as in the case of strings) use the **–l** flag to display the entire field.

Data input formatting

Binary data for attributes of binary type can be entered in the following formats:

- ″0x*nnnnnnnn* 0x*nnnnnnnn* 0x*nnnn*...″
- ″0x*nnnnnnnnnnnnnnnnnn*...″
- 0x*nnnnnnnnnnnnnnnn*...

Integer data for attributes of one of the integer types can be entered as:

- A decimal constant that begins with a non-zero digit (**Int32=45**, for example)
- An octal constant that begins with a prefix of **0**, which is optionally followed by a combination of decimal numbers in the range **0** to **7** (**Int32=055**, for example)
- A hexadecimal constant that begins with a prefix of **0x** or **0X** followed a combination of decimal numbers in the range **a** to **f** and **A** to **F** (**Int32=0x2d**, for example)

Be careful when you specify strings as input data. Strings that contain:

- No white space or non-alphanumeric characters can be entered as input without enclosing quotation marks
- White space or other alphanumeric characters must be enclosed in quotation marks
- Single quotation marks (') must be enclosed by double quotation marks (″), as shown in this example: `"this is a string with 'single quotation marks'"`

Selection strings must be enclosed in double quotation marks, unless the selection string itself contains double quotation marks, in which case the selection string must be enclosed in single quotation marks. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

- Sample selection string input: `"NodeNumber == 1"`

- Selection string input where double quotation marks are part of the selection string: `'Name == "c175n05"'`

Structured data (SD) types must be enclosed in square brackets: `[hello,1,{2,4,6,8}]`

When supplying structured data (SD) as command-line input to the RMC commands, enclose the SD in single quotation marks: `SD='[hello,1,{2,4,6,8}]'`

Arrays of any type must be enclosed in braces {}:

- Array of integers: `{-4, -3, -2, -1, 0, 1, 2, 3, 4}`
- Array of strings: `{abc, "do re mi", 123}`
- Array of structured data: `{[hello,1,{0,1,2,3}],[hello2,2,{2,4,6,8}]}`

Arrays of any type with more than one element must be enclosed in quotation marks. For example:

- mkrsrc IBM.Foo Name=testing NodeList={1} Uint32Array='{1,2,3}'
- mkrsrc IBM.Foo Name=testing NodeList='{1}' Uint32_array='{1,2,3}'

Arrays of strings and arrays of structured data must always be enclosed in quotation marks.

When supplying arrays of structured data or arrays containing strings enclosed in quotation marks as command-line input to the RMC commands, enclose the entire array in single quotation marks:

- Array of strings: `mkrsrc IBM.Foo Name="c175n05" NodeList={1} StringArray='{"a string","a different string"}'`
- Array of structured data: `mkrsrc IBM.Foo Name="c175n05" NodeList={1} SDArray='{["string 1",1,{1,1}],["string 2",2,{1,2,3}]}'`

For more examples, see the **resource_data_input** man page.

Data output formatting

String data is always displayed in either double or single quotation marks, as shown below:

- A description attribute that equals the string ″This is a string that contains white space″ is displayed using long format as:

  ```
  Description = "This is a string that contains white space"
  ```

- A description attribute value that equals an empty string ″″ is displayed in long format as:

  ```
  Description = ""
  ```

- A description attribute value that equals a string that contains a new-line character at the end of the string is displayed in long format as:

  ```
  Description = "This string ends with a new-line character..."
  ```

- A selection string containing double quotation marks is displayed in long format as:

  ```
  SelectionString = 'Name == "c175n05"'
  ```

- A name attribute value that equals the string ″c175n05″ is displayed in long format as:

  ```
  Name = "c175n05"
  ```

Binary data is displayed as follows:

"0x*nnnnnnnn* 0x*nnnnnnnn* 0x*nnnnnnnn* 0x*nnnnnnnn*"

Naming conventions

The following variable names are used throughout the RMC command man pages:

Variable	Description
attr	The name of a resource class or a resource attribute
resource_class	The name of a resource class

Node groups

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*.

Terminology

attribute

Attributes are either persistent or dynamic. A resource class is defined by a set of persistent and dynamic attributes. A resource is also defined by a set of persistent and dynamic attributes. Persistent attributes define the configuration of the resource class and resource. Dynamic attributes define a state or a performance-related aspect of the resource class and resource. In the same resource class or resource, a given attribute name can be specified as either persistent or dynamic, but not both.

resource

An entity in the system that provides a set of services. Examples of hardware entities are processors, disk drives, memory, and adapters. Examples of software entities are database applications, processes, and file systems. Each resource in the system has one or more attributes that define the state of the resource.

resource class

A broad category of system resource, for example: node, file system, adapter. Each resource class has a container that holds the functions, information, dynamic attributes, and conditions that apply to that resource class. For example, the “/tmp space used” condition applies to a file system resource class.

resource manager

A process that maps resource and resource-class abstractions into calls and commands for one or more specific types of resources. A resource manager can be a standalone daemon, or it can be integrated into an application or a subsystem directly.

To see all of the resource classes defined in the system, run the **lsrsrc** command without any flags or parameters. To see all of the resources defined in the system for the **IBM.FileSystem** resource class, enter:

```
lsrsrc IBM.FileSystem
```

selection string

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 can be listed in a selection string. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

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.

All RMC commands include a **-T** flag and a **-V** flag. Use the **-T** flag only when your software service organization instructs you to turn tracing on. Trace messages are not translated. Use the **-V** flag, which indicates ″verbose″ mode, to see more information about the command. Verbose messages are contained in message catalogs and are translated based on the locale in which you are running and other criteria.

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 man page is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/man/rmccli

Related information

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about RMC operations

Commands: **nodegrp**

“resource_data_input ” on page 70

Part 2. RSCT peer domains: creating and administering

Chapter 4. Configuration resource manager commands 83
addrpnode 84
chcomg 88
forcerpoffline 92
lscomg 94
lsrpdomain 98
lsrpnode 101
mkcomg 106
mkrpdomain 110
preprpnode 117
rmcomg 120
rmrpdomain 123
rmrpnode 126
startrpdomain 129
startrpnode 133
stoprpdomain 136
stoprpnode 139

© Copyright IBM Corp. 2002, 2007

Chapter 4. Configuration resource manager commands

© Copyright IBM Corp. 2002, 2007

addrpnode

Purpose

Adds one or more nodes to a peer domain definition.

Syntax

addrpnode [**–c**] [**–h**] [**–TV**] *node_name1* [*node_name2* ...]

addrpnode [**–c**] { **–f** | **–F** { *node_file* | *"–"* } } [**–h**] [**–TV**]

Description

Before running addrpnode:
To set up the proper security environment, run the **preprpnode** command on each node that is to be added to the peer domain.

The **addrpnode** command adds the specified nodes to the online peer domain in which it (**addrpnode**) is run. This command must be run on a node that is online to the peer domain in which the new nodes are to be added. Though a node can be defined in multiple peer domains, it can only be online in one peer domain. To add one or more nodes to the peer domain, more than half of the quorum nodes must be online.

To enable **addrpnode** to continue when there is an error on one of the nodes, use the **-c** flag.

The **addrpnode** command does *not* bring the added nodes online in the peer domain. To do this, use the **startrpnode** command.

Parameters

node_name1 **[***node_name2* **...]**
Specifies one or more nodes that are to be added to the 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.

Flags

–c Continues processing the command as long as at least one node can be added to the peer domain.

By default, if the **addrpnode** command fails on any node, it will fail on all nodes. The **-c** flag overrides this behavior, so that the **addrpnode** command will run on the other nodes, even if it fails on one node.

–f | –F { *node_file* **|** *"–"* **}**
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.

- Specify one 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.

By default, all of the nodes listed in *node_file*:

1. are group services group leader candidates
2. are used for quorum decisions

Here's an example of a basic node file:

```
# peer domain nodes for addrpnode
nodeA       # production node
nodeB       # test node
nodeC       # production node
nodeD       # test node
```

To explicitly specify that a node is a group services group leader candidate, add the **p** (or **P**) option, preceded by an at sign (**@**) control character, after the node name and before any comments. For example:

```
nodeC   @P       # production node
```

To *exclude* the node from being a group services group leader candidate, use **@!P**. For example:

```
nodeD   @!P      # test node
```

To explicitly specify that a node is to be used for quorum decisions, use the **q** (or **Q**) option. For example:

```
nodeA   @Q       # production node
```

To *exclude* the node from being used for quorum decisions, use **@!Q**. For example:

```
nodeB   @!Q      # test node
```

Use only one **@** control character per line. If you use a node number, add it after the node name and before any comments. The node number can be before or after the quorum and group leader specifications. For example:

```
nodeA       @PQ        # group leader candidate, used for quorum decisions
nodeB       @P!Q       # group leader candidate, not used for quorum decisions
nodeC       @!PQ   3   # used for quorum decisions, not a group leader candidate
nodeD   4   @!P!Q      # not used for quorum decisions, not a group leader candidate
```

When specifying **P**s and **Q**s in a node file, consider the following (where *x* is **P** or **Q**):

- 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 *RSCT: Administration Guide* for more information.

Use **-f** ″**-**″ or **-F** ″**-**″ to read the node names from standard input.

-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.

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.

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.

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.

Security

The user of the **addrpnode** command needs write permission for the **IBM.PeerDomain** resource class and the **IBM.PeerNode** resource class on each node that is to be added to the peer domain. This is set up by running the **preprpnode** command on each node to be added. Specify the names of all the nodes online in the peer domain with the **preprpnode** command. This gives the online nodes the necessary authority to perform operations on the nodes to be added.

Restrictions

This command must be run on a node that is online in the peer domain in which the new nodes are to be added.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/addrpnode

Examples

1. To add the nodes **nodeB** and **nodeC** to the peer domain **ApplDomain** where **nodeA** is already defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   addrpnode nodeB nodeC
   ```

2. In this example, **nodeA** is already defined and online in the **ApplDomain** peer domain. To add **nodeB**, **nodeC**, **nodeD**, and **nodeE** to **ApplDomain** using the **/pd/pdnodes.config** file, run this command on **nodeA**:

   ```
   addrpnode -f /pd/pdnodes.config
   ```

   where the contents of **/pd/pdnodes.config** are:

   ```
   # peer domain nodes for addrpnode
   nodeB       # dev node
   nodeC       # prod node
   nodeD       # test node
   nodeE       # test node
   ```

3. In this example, **nodeA** is already defined and online in the **ApplDomain** peer domain. To add **nodeB**, **nodeC**, **nodeD**, and **nodeE** to **ApplDomain** using the **/pd/pdnodes.config** file, specifying that **nodeC** and **nodeD** can be used for quorum decisions and **nodeD** cannot be a group services group leader, run this command on **nodeA**:

   ```
   addrpnode -f /pd/pdnodes.config
   ```

   where the contents of **/pd/pdnodes.config** are:

   ```
   # peer domain nodes for addrpnode
   nodeB            # dev node
   nodeC   @Q       # prod node
   nodeD   @Q!P     # test node
   nodeE            # test node
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmrpnode**, **startrpdomain**, **startrpnode**

chcomg

Purpose

Changes a previously-defined communication group for a peer domain.

Syntax

To change an attribute of a communication group:

chcomg [**–s** *sensitivity*] [**–p** *period*] [**–t** *priority*] [**–b**] [**–r**] [**–x b** | **r** | **br**] [**–e** *NIM_path*] [**–m** *NIM_parameters*] [**–h**] [**–TV**] *communication_group*

To change a reference in an interface resource to a different communication group:

chcomg [**–i n:***network_interface1*[**:***node1*][**,***network_interface2*[**:***node2*]...] | **–S n:***"network_interface_selection_string"*] [**–h**] [**–TV**] *communication_group*

Description

The **chcomg** command changes an existing communication group definition with the name specified by the *communication_group* parameter for the online peer domain. 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.

The **chcomg** command must be run on a node that is currently online in the peer domain where the communication group is defined. One or more attributes can be changed with one **chcomg** command, but at least one change is required.

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.

The **chcomg** command can also be used to assign a communication group to an interface resource. 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. Before a communication group can be removed, any interface resources that refer to it must be reassigned.

| More than half of the quorum nodes must be online to change a communication group in the domain.

Parameters

communication_group
: Specifies the name of an existing communication group to be changed in the peer domain.

Flags

-s *sensitivity*
Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity is an integer that is greater than or equal to **4**.

-p *period*
Specifies the period, which is the number of seconds between heartbeats. The value of *period* can be an integer or a floating-point number that is greater than or equal to **1**.

-t *priority*
Specifies the priority. The priority indicates the importance of this communication group with respect to others. It is used to order the heartbeat rings. The lower the number, the higher the priority. The highest priority is **1**.

-b Specifies that broadcast will be used if the underlying media support it. The **-b** flag cannot be used when specifying **-x b**.

-r Specifies that source routing will be used if the underlying media support it. The **-r** flag cannot be used when specifying **-x r**.

–x b | r | br
Excludes control for the heartbeat mechanism. This indicates that one or more controls for heartbeat mechanisms should not be used even if the underlying media support it. The following can be excluded:

b Specifies that broadcast should not be used even if the underlying media support it.

r Specifies that source routing should not be used even if the underlying media support it.

Excluding more than one control is specified by listing the feature option letters consecutively (**-x br**).

-i n:*network_interface1***[:***node1***] [,***network_interface2***[:***node2***]...**
Assigns this communication group to the network interface resource defined by the network interface resource name and optionally the node name where it can be found.

If **-i** is specified, **-S** cannot be specified.

-S n: *"network_interface_selection_string"*
Assigns this communication group to the interface specified by the network interface selection string.

If **-S** is specified, **-i** cannot be specified.

-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 is a character string that is passed to the NIM when starting it.

-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.

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.

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.

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.

Security

The user of the **chcomg** 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.

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 changed.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chcomg

Examples

In these examples, node **nodeA** is defined and online to peer domain **ApplDomain**.

1. To change the communication group **ComGrp1** for **ApplDomain** to a sensitivity of 4 and period of 3, run this command on **nodeA**:

   ```
   chcomg -s 4 -p 3 ComGrp1
   ```

2. To change the communication group **ComGrp1** for **ApplDomain** to use broadcast, run this command on **nodeA**:

   ```
   chcomg -b ComGrp1
   ```

3. To change the communication group **ComGrp1** for **ApplDomain** to no longer use source routing, run this command on **nodeA**:

   ```
   chcomg -x r ComGrp1
   ```

4. To change the communication group **ComGrp1** for **ApplDomain**, to use a NIM path of **/usr/sbin/rsct/bin/hats_nim**, and to use NIM parameters **-l 5** to set the logging level, run this command on **nodeA**:

   ```
   chcomg -e /usr/sbin/rsct/bin/hats_nim -m "-l 5" ComGrp1
   ```

5. To assign the communication group **ComGrp1** for **ApplDomain** to the network interface resource named **eth0** on **nodeB**, run this command on **nodeA**:

   ```
   chcomg -i n:eth0:nodeB ComGrp1
   ```

6. To assign the communication group **ComGrp1** for **ApplDomain** to the network interface resource that uses the subnet 9.123.45.678, run this command on **nodeA**:

   ```
   chcomg -S n:"Subnet == '9.123.45.678'" ComGrp1
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lscomg**, **lsrpdomain**, **lsrpnode**, **mkcomg**, **preprpnode**, **rmcomg**

forcerpoffline

Purpose

Forces a peer domain offline.

Syntax

forcerpoffline [**–h**] *domain_name*

Description

Use this command with extreme caution.

The **forcerpoffline** command should only be used if a node is in a ″pending online″ state and you are unable to bring it online using **startrpdomain**. This could occur if you try to bring the node online while the domain is operating under quorum. If you are not sure why the node is stuck in the ″pending online″ state, run the **ctsnap** command before using **forcerpoffline**. As a result of running **forcerpoffline**, the configuration resource manager subsystem (**IBM.ConfigRM**) and the RMC subsystem (**ctrmc**) will be recycled.

Parameters

domain_name Specifies the name of a previously-defined peer domain that is to be forced offline.

Flags

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

Files

The **/var/ct/cfg/current_cluster** file and the **/var/ct/cfg/default_cluster** file are modified.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Exit status

0 The command ran successfully.

1 The command terminated due to an underlying RMC error.

2 The command terminated due to an underlying error in the command script.

3 The command terminated because the user specified a non-valid flag.

4 The command terminated because the user specified a non-valid parameter.

5 The command terminated due to a user error (specifying a domain name that does not exist, for example).

Security

You must have **root** authority to run this command.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/forcerpoffline

Related information

Commands: **ctsnap**, **startrpdomain**, **startrpdomain**

lscomg

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*. If you specify the **-i** flag, **lscomg** 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*

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.

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.

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.

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.

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.

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.

Restrictions

This command must be run on a node that is defined and online to the peer domain on which the communcation group exists.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lscomg

Examples

In these examples, **nodeA** is defined and online to peer domain **ApplDomain**.

1. To display general information about the communication groups for **ApplDomain**, run this command on **nodeA**:

   ```
   lscomg
   ```

   The output will look like this:

```
Name  Sensitivity  Period  Priority  Broadcast  SourceRouting  NIMPath                     NIMParameters
ComG1      2          2        1        no          yes        /usr/sbin/rsct/bin/hats_nim      -l 5
```

2. To display information about the interface resources that refer to the communication group **ComGrp1** for the peer domain **ApplDomain**, run this command on **nodeA**:

   ```
   lscomg -i ComGrp1
   ```

   The output will look like this:

```
Name    NodeName    IPAddr          SubnetMask       Subnet
eth0    n24         9.234.32.45     255.255.255.2    9.235.345.34
eth0    n25         9.234.32.46     255.255.255.2    9.235.345.34
```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **chcomg**, **lsrpdomain**, **lsrpnode**, **mkcomg**, **preprpnode**, **rmcomg**

lsrpdomain

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

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.

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.

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.

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.

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.

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.

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 Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsrpdomain

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
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmrpdomain**, **startrpdomain**, **stoprpdomain**

lsrpnode

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] [**–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.

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:

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), the following field is displayed when you specify the **-P** flag:

Preferred	Indicates whether the node is a group services group leader candidate. See the *RSCT: Administration Guide* for information about group services group leader selection.

Along with other fields (depending on the flags specified), the following field is displayed when you specify the **-Q** flag:

Quorum Indicates whether the node participates in quorum decisions. See the *RSCT: Administration Guide* for information about quorum decisions.

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.

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.

-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 service group leader. See the *RSCT: Administration Guide* 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 *RSCT: Administration Guide* for more information on quorum decisions.

-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.

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.

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.

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.

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.

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.

Examples

1. To display general information about the nodes in the online peer domain that **nodeA** belongs to, run this command on **nodeA**:

   ```
   lsrpnode
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion
   nodeA   Online    2.5.0.0
   nodeB   Online    2.5.0.0
   nodeC   Offline   2.5.0.0
   ```

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:2.5.0.0:
   nodeB:Online:2.5.0.0:
   nodeC:Offline:2.5.0.0:
   ```

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 = 2.5.0.0

   Name        = nodeB
   OpState     = Online
   RSCTVersion = 2.5.0.0

   Name        = nodeC
   OpState     = Offline
   RSCTVersion = 2.5.0.0
   ```

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    2.5.0.0       2         40a514bed9d82412
   nodeB   Online    2.5.0.0       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    2.5.0.0       yes
   nodeB   Online    2.5.0.0       yes
   nodeC   Online    2.5.0.0       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    2.5.0.0       no
   nodeB   Online    2.5.0.0       yes
   nodeC   Online    2.5.0.0       yes
   ```

Location

/usr/sbin/rsct/bin/lsrpnode

Related information

"rmccli " on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **preprpnode**, **startrpnode**, **stoprpnode**

mkcomg

Purpose

Creates a new communication group definition for a peer domain.

Syntax

mkcomg [**–s** *sensitivity*] [**–p** *period*] [**–t** *priority*] [**–x b** | **r** | **br**] [**–e** *NIM_path*] [**–m** *NIM_parameters*] [**–i n:***network_interface1*[**:***node1*] [**,***network_interface2*[**:***node2*]...] | **–S n:***"network_interface_selection_string"*] [**–h**] [**–TV**] *communication_group*

Description

The **mkcomg** command creates a new communication group definition for an online peer domain with the name specified by the *communication_group* parameter. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The communication group determines which devices are used for heartbeating in the peer domain. There can be more than one communication group in a peer domain.

The **mkcomg** command must be run on a node that is currently online in the peer domain where the communication group is to be defined. More than half of the quorum 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.

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.

Flags

-s *sensitivity*
: Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity value is an integer greater than or equal to 2. The default value is 4.

-p *period*
: Specifies the number of seconds between heartbeats. The period is an integer greater than or equal to 1. The default value is 1.

-t *priority*
Specifies the priority. This value indicates the importance of this communication group with respect to others. It is used to order the heartbeat rings. The lower the number means the higher the priority. The highest priority is 1. The default value is 1 for IP networks and 255 for RS232 networks.

–x 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**.

-e *NIM_path*
Specifies the network interface module (NIM) path name. This character string specifies the path name to the NIM that supports the adapter types in the communication group.

-m *NIM_parameters*
Specifies the NIM start parameters. This character string is passed to the NIM when starting it.

-i n:*network_interface1***[:***node1***] [,***network_interface2***[:***node2***]...**
Assigns this communication group to the network interface resource defined by the network interface resource name and optionally the node name where it can be found.

If **-i** is specified, **-S** cannot be specified.

-S n:*"network_interface_selection_string"*
Assigns this communication group to the interface specified by the network interface selection string.

If **-S** is specified, **-i** cannot be specified.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

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.

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.

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.

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.

Location

/usr/sbin/rsct/bin/mkcomg

Examples

1. To define the communication group **ComGrp1** for the peer domain **ApplDomain** and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg ComGrp1
   ```

2. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, using a sensitivity of 1 and period of 3, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -s 1 -p 3 ComGrp1
   ```

3. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, not using broadcast, using a priority of 3, and **nodeA** is defined and online to **ApplDomain**, run this command on nodeA:

   ```
   mkcomg -x b -t 3 ComGrp1
   ```

4. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, not using broadcast, not using source routing, and **nodeA** is defined and online to **ApplDomain**, run the following command on **nodeA**:

   ```
   mkcomg -x br ComGrp1
   ```

5. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, using a NIM path of **/usr/sbin/rsct/bin/hats_nim**, NIM parameters **-l 5** to set the logging level, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -e /usr/sbin/rsct/bin/hats_nim -m "-l 5" ComGrp1
   ```

6. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, assign **ComGrp1** to the network interface resource named **eth0** on **nodeB**, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -i n:eth0:nodeB ComGrp1
   ```

7. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, assign **ComGrp1** to the network interface resource that uses the subnet 9.123.45.678, and **nodeA** is defined and online to **ApplDomain**, run this command on **nodeA**:

   ```
   mkcomg -S n:"Subnet == 9.123.45.678" ComGrp1
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **lscomg**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmcomg**, **startrpdomain**

mkrpdomain

Purpose

Creates a new 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*] [**–k** *cssk_type* [**–r** *refresh_interval*]] [**–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*] [**–k** *cssk_type* [**–r** *refresh_interval*]] [**–h**] [**–TV**] *peer_domain*

Description

The **mkrpdomain** command creates a new peer domain definition with the name specified by the *peer_domain* parameter. The nodes that are specified on the command line or listed in *file_name* are defined to the new peer domain. A peer domain can be used to provide high-availability services when configuring application and system resources.

The **preprpnode** command must have been run on each of the nodes to be defined to the peer domain. The **preprpnode** command prepares the security environment for the peer domain operations. See the **preprpnode** command for more information about peer domain definition requirements. Only those nodes that have the appropriate security setup will be successfully defined to the peer domain.

If the UDP port numbers for group services and topology services are not available on all of the nodes to be defined to the peer domain, the **mkrpdomain** command will fail. The command will also fail if the peer domain name is already being used or if any node cannot be successfully 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 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 starting the peer domain. 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
```

In peer domains with an RSCT active version of 2.4.7.1 or later, 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 will affect 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 two 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 two minutes of each other, messages passed between a sending node and a receiving node with a time difference that is longer than two minutes will be discarded.

When message authentication is enabled using the **-k** flag, a key refresh interval can be specified 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 *RSCT: Administration Guide*.

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 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.

Parameters

peer_domain
: Specifies the name of the new peer domain to be created. You can only use these ASCII characters 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 hostname. The node name must resolve to an IP address.

Flags

-t *TS_port*
: Specifies the topology services port number. This UDP port is used for daemon-to-daemon communication. Any unused port in the range 1024 to 65535 can be assigned. The command will fail if the specified port is unavailable. The default is 12347.

-g *GS_port*
: Specifies the group services port number. This UDP port is for

daemon-to-daemon communication. Any unused port in the range 1024 to 65535 can be assigned. The command will fail if the specified port is unavailable. The default is 12348.

–c Continues to run the **mkrpdomain** command on the remaining nodes.

By default, if the **mkrpdomain** command fails on any node, it will fail on all nodes. The **-c** flag overrides this behavior, so that the **mkrpdomain** command will run on the other nodes, even if it fails on one node.

-Q *quorum_type* **|** *quorum_type_name*
Specifies the quorum rules that are used for start-up, operational, and configuration quorum. Start-up quorum defines how many quorum nodes are contacted to obtain configuration information before starting the peer domain. Operational quorum defines how many quorum nodes must be online in order to start and stop resources and how tie breaking is used. Configuration quorum defines how many quorum nodes must be online to make changes to 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:

0 | normal
Specifies normal quorum rules. This is the default. For start-up quorum, at least half of the quorum nodes will be contacted for configuration information. For configuration quorum, more than half of the quorum nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the quorum 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 start-up quorum, even if no other nodes can be contacted, the node will still come 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.

-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.

–f | –F { *node_file* **|** *"–"* **}**
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.

- Specify one 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.

By default, all of the nodes listed in *node_file*:

1. are group services group leader candidates

2. are used for quorum decisions

Here's an example of a basic node file:

```
# peer domain nodes for addrpnode
nodeA       # production node
nodeB       # test node
nodeC       # production node
nodeD       # test node
```

To explicitly specify that a node is a group services group leader candidate, add the **p** (or **P**) option, preceded by an at sign (**@**) control character, after the node name and before any comments. For example:

```
nodeC   @P       # production node
```

To *exclude* the node from being a group services group leader candidate, use **@!P**. For example:

```
nodeD   @!P      # test node
```

To explicitly specify that a node is to be used for quorum decisions, use the **q** (or **Q**) option. For example:

```
nodeA   @Q       # production node
```

To *exclude* the node from being used for quorum decisions, use **@!Q**. For example:

```
nodeB   @!Q      # test node
```

Use only one **@** control character per line. If you use a node number, add it after the node name and before any comments. The node number can be before or after the quorum and group leader specifications. For example:

```
nodeA       @PQ        # group leader candidate, used for quorum decisions
nodeB       @P!Q       # group leader candidate, not used for quorum decisions
nodeC       @!PQ   3   # used for quorum decisions, not a group leader candidate
nodeD   4   @!P!Q      # not used for quorum decisions, not a group leader candidate
```

When specifying **P**s and **Q**s in a node file, consider the following (where *x* is **P** or **Q**):

- 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 *RSCT: Administration Guide* for more information.

Use **-f** ″**-**″ or **-F** ″**-**″ to read the node names from standard input.

-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, the stronger the encryption algorithm. The stronger the algorithm, the slower the performance. The valid key types are:

CSSKTYPE_None
Indicates that message authentication is disabled. This is the default.

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 the most data protection, but with slower performance and more data overhead.

This flag is valid in peer domains with an RSCT active version of 2.4.7.1 or later.

-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 one day. The minimum refresh interval is 30 seconds. The maximum refresh interval is 30 days.

The **-r** flag can only be specified when the **-k** flag is used.

This flag is valid in peer domains with an RSCT active version of 2.4.7.1 or later.

-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.

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.

Files

The **/etc/services** file is modified.

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.

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.

Security

The user of the **mkrpdomain** command needs write permission to the **IBM.PeerDomain** resource class on each node that is to be defined to the peer domain. This is set up by running the **preprpnode** command on each node that is to be defined to the domain, specifying the name of the node on which the user will run **mkrpdomain**.

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) fileset for AIX.

Location

/usr/sbin/rsct/bin/mkrpdomain

Examples

1. To define a peer domain called **ApplDomain** that consists of a node called **nodeA**, run this command on **nodeA**:

   ```
   mkrpdomain ApplDomain nodeA
   ```

2. To define a peer domain called **ApplDomain** that consists of three nodes called **nodeA**, **nodeB**, and **nodeC**, run this command on **nodeA**, **nodeB**, or **nodeC**:

   ```
   mkrpdomain ApplDomain nodeA nodeB nodeC
   ```

3. To define a peer domain called **ApplDomain** that consists of two nodes 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 called **ApplDomain** that consists of two nodes called **nodeA** and **nodeB** 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 called **ApplDomain** that consists of the nodes **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, using the the **/pd/pdnodes.config** file, run this command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of /pd/pdnodes.config are:

   ```
   # 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 called **ApplDomain** that consists of the nodes **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, using the **/pd/pdnodes.config** file, which specifies that **nodeB** and **nodeC** cannot be used in quorum decisions and **nodeC** and **nodeD** cannot be the group services group leader, run this command on any of the nodes:

   ```
   mkrpdomain -f /pd/pdnodes.config ApplDomain
   ```

   where the contents of /pd/pdnodes.config are:

   ```
   # peer domain nodes for mkrpdomain
   nodeA           # dev node
   nodeB    @!Q    # dev node
   nodeC    @!Q!P  # prod node
   nodeD    @!P    # test node
   nodeE           # test node
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations, including setting up and managing CSSK settings

Commands: **addrpnode**, **chrsrc**, **lsrpdomain**, **lsrpnode**, **lsrsrc**, **preprpnode**, **rmrpdomain**, **rmrpnode**, **runact**, **startrpdomain**, **stoprpdomain**

preprpnode

Purpose

Prepares a node to be defined to a peer domain.

Syntax

preprpnode [**–k**] [**–h**] [**–TV**] *node_name1* [*node_name2* ...]

preprpnode –f | **–F** { *file_name* | *"–"* } [**–k**] [**–h**] [**–TV**]

Description

The **preprpnode** command prepares security on the node on which the command is run so it can be defined in a peer domain. It allows for peer domain operations to be performed on this node and must be run before the node can join a peer domain using the **mkrpdomain** or **addrpnode** command.

Before the **mkrpdomain** command is issued on a node, the **preprpnode** command must be run on each node to be defined to the new peer domain, using the name of the node that is to run the **mkrpdomain** command as the parameter. This gives the **mkrpdomain** node the necessary authority to create the peer domain configuration on each new node and set up additional security.

Before the **addrpnode** command is issued on a node, the **preprpnode** command must be run on each node that is to be added, using the names of all online nodes as the parameters. This gives the online nodes the authority to perform the necessary operations on the new node.

The **preprpnode** command performs the following:

1. Establishes trust with the node names specified on the command by adding their public keys to the trusted host list.
2. Modifies the resource monitoring and control (RMC) access control list (ACL) file to enable access to peer domain resources on this node from the other nodes in the peer domain. This allows peer domain operations to occur on the node. The RMC subsystem is refreshed so that these access changes will take effect.
3. RMC remote connections are enabled.

If the nodes that are to be defined to a peer domain are already in a management domain, you do not need to exchange public keys. You can use the **-k** flag to omit this step.

Parameters

node_name1 **[***node_name2* **...]**

Specifies the node (or nodes) from which peer domain commands can be accepted. Typically, this is the name of the node that will be running the **mkrpdomain** command when forming the peer domain. When adding to the peer domain, it is a list of the nodes that are currently online in the peer domain. 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.

Flags

–f | –F { *file_name* **|** *"–"* **}**
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

Use **-f** *"-"* or **-F** *"-"* to specify **STDIN** as the input file.

–k Specifies that the command should not exchange public keys.

–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.

Files

The access control list (ACL) file — **/var/ct/cfg/ctrmc.acls** — is modified. If this file does not exist, it is created.

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.

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.

Security

The user of the **preprpnode** command needs write permission to the access control list (ACL) file. Permissions are specified in the ACL file. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Restrictions

This command must run on a node that will be defined to the peer domain.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/preprpnode

Examples

1. Suppose **mkrpdomain** will be issued from **nodeA**. To prepare **nodeB**, **nodeC**, and **nodeD** to be defined to a new peer domain, **AppIDomain**, run this command on **nodeB**, on **nodeC**, and then on **nodeD**:

   ```
   preprpnode nodeA
   ```

2. Suppose **nodeA** and **nodeB** are online in **AppIDomain**. To prepare **nodeC** to be added to the existing domain, run this command on **nodeC**:

   ```
   preprpnode nodeA nodeB
   ```

   Alternatively, create a file called **onlineNodes** with these contents:

   ```
   nodeA
   nodeB
   ```

   Then, run this command on **nodeC**:

   ```
   preprpnode -f onlineNodes
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**

Files: **ctrmc.acls**

rmcomg

Purpose

Removes a communication group that has already been defined from a peer domain.

Syntax

rmcomg [**–q**] [**–h**] [**–TV**] *communication_group*

Description

The **rmcomg** command removes the definition of the existing communication group with the name specified by the *communication_group* parameter for the online peer domain. 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.

The **rmcomg** command must be run on a node that is currently online in the peer domain where the communication group is defined. More than half of the quorum nodes must be online to remove a communication group from the domain.

The communication group must not be referred to by an interface resource. Use the **chcomg** command to remove references made by interface resources to a communication group.

Parameters

communication_group
: Specifies the name of the defined communication group that is to be removed from the peer domain.

Flags

-q Specifies quiet mode. The command does not return an error if the communication group does not exist.

-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.

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.

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.

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 does not exist.

Security

The user of the **rmcomg** command needs write permission for the **IBM.CommunicationGroup** resource class. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

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 removed.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmcomg

Examples

In this example, **nodeA** is defined and online to **ApplDomain**. To remove the communication group definition **ComGrp1** for the peer domain **ApplDomain**, run this command on **nodeA**:

```
rmcomg ComGrp1
```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **chcomg**, **lscomg**, **lsrpdomain**, **lsrpnode**, **mkcomg**, **preprpnode**

rmrpdomain

Purpose

Removes a peer domain that has already been defined.

Syntax

rmrpdomain [**–f**] [**–q**] [**–h**] [**–TV**] *peer_domain*

Description

The **rmrpdomain** command removes the peer domain definition that is specified by the *peer_domain* parameter. The peer domain that is to be removed must already be defined. This command must be run on a node that is defined in the peer domain. When **rmrpdomain** is run on a node that is online to the peer domain, it removes the peer domain definition on all nodes defined to the peer domain that are reachable from that node. If a node defined to the peer domain is not reachable, that node's local peer domain definition is not removed. To remove the local peer domain definition when the peer domain is not online or when the node is not online to the peer domain, run the **rmrpdomain** command on that node and specify the **-f** flag.

The most efficient way to remove a peer domain definition is to make sure the peer domain is online. Then, from a node that is online to the peer domain, run the **rmrpdomain** command. If there are nodes that are not reachable from the node on which the **rmrpdomain** command was run, on each of those nodes, run the **rmrpdomain** command using the **-f** flag. This can be done at a later time if the node itself is not operational.

The **-f** flag must also be used to override a subsystem's rejection of the peer domain removal. A subsystem may reject the request if a peer domain resource is busy, for example. Specifying the **-f** flag in this situation indicates to the subsystems that the peer domain definition must be removed.

The **rmrpdomain** command does not require configuration quorum. Therefore, this command is still successful if it is issued to a minority sub-cluster. Later, the majority sub-cluster may become active. If so, the domain is still removed.

Parameters

peer_domain Specifies the name of the defined peer domain that is to be removed.

Flags

-f Forces the peer domain to be removed. The force flag is required to remove a peer domain definition:

- from the local node when the node is not online to the peer domain.
- when a subsystem may reject the request, as when resources are allocated, for example.

-q Specifies quiet mode. The command does not return an error if the peer domain does not exist.

-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.

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.

Files

The **/etc/services** file is modified.

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.

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.

Security

The user of the **rmrpdomain** command needs write permission to the **IBM.PeerDomain** resource class on each node that is to be defined to the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Restrictions

The node on which this command is run must be defined to the peer domain and should be able to reach all of the nodes that are defined to the peer domain. The node's local peer domain definition will not be removed if the node is not reachable.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmrpdomain

Examples

1. To remove the peer domain definition of **ApplDomain** where **nodeA**, **nodeB**, and **nodeC** are defined and online to *ApplDomain*, and all are reachable to each other, run this command on *nodeA*, **nodeB**,or **nodeC**:

   ```
   rmrpdomain ApplDomain
   ```

2. To remove the local peer domain definition of **ApplDomain** on **nodeD** when **nodeD** is not online to the peer domain, the peer domain is offline, or the peer domain does not exist, run this command on **nodeD**:

   ```
   rmrpdomain -f ApplDomain
   ```

3. To remove the peer domain definition of **ApplDomain** where **nodeA**, **nodeB**, and **nodeC** are defined and online to **ApplDomain**, all are reachable to each other, and to prevent a subsystem from rejecting the request, run this command on **nodeA**, **nodeB**, or **nodeC**:

   ```
   rmrpdomain -f ApplDomain
   ```

Related information

"rmccli " on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **rmrpnode**, **startrpdomain**, **stoprpdomain**

Files: **/etc/services**

rmrpnode

Purpose

Removes one or more nodes from a peer domain definition.

Syntax

rmrpnode [**–f**] [**–q**] [**–h**] [**–TV**] *node_name1* [*node_name2* ...]

rmrpnode –F { *file_name* | *"–"* } [**–f**] [**–q**] [**–h**] [**–TV**]

Description

The **rmrpnode** command removes one or more nodes from the online peer domain where the command is run. The command must be run on a node that is online to the peer domain in which the nodes are to be removed. The nodes that are to be removed must be offline to the peer domain and must be reachable from the node where the command is run. To take nodes offline, use the **stoprpnode** command.

Specifying the **-f** flag forces the specified nodes to be removed from the peer domain. If the **-f** flag is not specified, more than half of the quorum nodes must be online to remove one or more nodes from the domain.

Parameters

node_name1 **[***node_name2* **...]**
Specifies the peer domain node names of the nodes to be removed from the peer domain definition. You can remove one or more nodes using the **rmrpnode** command. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Flags

-f Forces the specified nodes to be removed from the peer domain.

-q Specifies quiet mode. The command does not return an error if the specified nodes are not in the peer domain.

–F { *file_name* **|** *"–"* **}**
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

Use **-F** *"-"* to specify **STDIN** as the input file.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

Standard input

When the **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

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 node does not exist in the peer domain.

Security

The user of the **rmrpnode** command needs write permission for the **IBM.PeerNode** resource class on each node that is to be removed from the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Restrictions

This command must be run on a node that is online in the peer domain in which the nodes are to be removed. The nodes to be removed must also be offline to the peer domain.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmrpnode

Examples

To remove the peer domain definitions of nodes **nodeB** and **nodeC** from the peer domain **ApplDomain**, when **nodeA** is defined and online to **ApplDomain**, and **nodeB** and **nodeC** are reachable from **nodeA**, run this command from **nodeA**:

```
rmrpnode nodeB nodeC
```

Related information

"rmccli " on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpnode**, **preprpnode**, **startrpnode**, **stoprpnode**

startrpdomain

Purpose

Brings a peer domain that has already been defined online.

Syntax

startrpdomain [**–A** | **–L**] [**–t** *timeout*] [**–Q** *quorum_type* | *quorum_type_name*] [**–m** *fanout*] [**–h**] [**–TV**] *peer_domain*

Description

The **startrpdomain** command brings a defined peer domain online by starting the resources on each node belonging to the peer domain.

The **startrpdomain** command must be run on a node that is defined to the peer domain. The command invites all offline nodes defined to the peer domain to come online in the peer domain every time the command is run for the peer domain. The command can be run more than once in the peer domain. If all the nodes defined in the peer domain are already online, no action is performed.

The **startrpdomain** command determines the peer domain configuration to use to bring the peer domain online by examining the peer domain configuration on the nodes defined to the peer domain. The latest version of the peer domain configuration information that is found is used to bring the peer domain online. By default, the latest version of the peer domain configuration found on at least half of the quorum nodes is used. Specifying the **-A** flag causes the latest version of the peer domain configuration found on all of the nodes defined in the peer domain to be used. Specifying the **-L** flag causes the configuration on the local node to be used.

In determining the latest version of the peer domain configuration information, a configuration timeout defines when to stop checking versions and begin to bring the peer domain online. The default timeout value is 120 seconds. The timeout value can be changed using the **-t** flag. The timeout value should be at least long enough so that the latest version of the peer domain configuration information from at least half of the quorum nodes can be found.

A node can only be online to one peer domain at a time. The **startrpdomain** command cannot be run on a node for a peer domain when another peer domain is already online for that node.

Parameters

peer_domain Specifies the name of a previously-defined peer domain that is to be brought online.

Flags

-A Finds and uses the latest version of the peer domain configuration information from all of the quorum nodes in the peer domain. This flag cannot be specified if the **-L** flag is specified. If neither flag (**-A** or **-L**) is specified, the latest version of the peer domain configuration information from at least half of the quorum nodes in the peer domain is used.

-L Uses the latest version of the peer domain configuration information that is on the local node. This flag cannot be specified if the **-A** flag is specified. If

neither flag (**-A** or **-L**) is specified, the latest version of the peer domain configuration information from at least half of the quorum nodes in the peer domain is used.

-t *timeout*
Specifies the timeout value in seconds. This flag limits the amount of time used to find the latest version of the peer domain configuration. When the timeout value is exceeded, the latest version of the peer domain configuration information found thus far is used. The timeout value should be long enough so that the latest version of the peer domain configuration information from at least half of the quorum nodes can be found. The default timeout value is 120 seconds.

-Q *quorum_type* | *quorum_type_name*
Enables you to override the startup quorum mode. This can be specified as an integer quorum type or quorum type name. If you do not specify this flag, startup quorum mode will be specified using the **mkrpdomain** command's **-Q** flag (or the default quorum mode for your environment) when you created the peer domain. You can override the quorum startup mode only if the quorum mode has been defined as **normal** or **quick**. The valid values are:

0 | **normal**
Specifies normal start-up quorum rules. Half of the quorum nodes will be contacted for configuration information.

1 | **quick**
Specifies quick start-up quorum rules. One node will be contacted for configuration information.

-m *fanout*
Specifies the maximum number of threads to use for this start operation. The **-m** flag overrides the default *fanout* value for the specified peer domain. This value is stored as a persistent attribute in the peer domain's **IBM.PeerNode** class. *fanout* can be an integer from **16** to **2048**.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

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.

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.

Security

The user of the **startrpdomain** command needs write permission for the **IBM.PeerDomain** resource class on each node that is defined to the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Restrictions

This command must be run from a node that is defined to the peer domain.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/startrpdomain

Examples

In these examples, **nodeA** is one of the nodes defined to **ApplDomain**.

1. To bring **ApplDomain** online, run this command on **nodeA**:

   ```
   startrpdomain ApplDomain
   ```

2. To bring **ApplDomain** online using all of the nodes in the peer domain to obtain the latest version of the peer domain configuration information, run this command on **nodeA**:

   ```
   startrpdomain -A ApplDomain
   ```

3. To bring **ApplDomain** online using a peer domain configuration timeout value of 240 seconds (to make sure that at least half of the quorum nodes in the peer domain are used), run this command on **nodeA**:

   ```
   startrpdomain -t 240 ApplDomain
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **forcerpoffline**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **stoprpdomain**

startrpnode

Purpose

Brings one or more nodes online to a peer domain.

Syntax

startrpnode [**–h**] [**–TV**] *node_name1* [*node_name2* ...]

startrpnode –f | **–F** { *file_name* | *"–"* } [**–h**] [**–TV**]

Description

The **startrpnode** command brings one or more offline nodes online to a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The node that is being brought online must have already been defined to be in this peer domain using the **addrpnode** command or the **mkrpdomain** command. The node must not be online to any other peer domain.

Parameters

node_name1 **[***node_name2* **...]**
Specifies the peer domain node names of the nodes to be brought online to the peer domain. You can bring one or more nodes online using the **startrpnode** command. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Flags

–f | –F { *file_name* **|** *"–"* **}**
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

Use **-f** *"-"* or **-F** *"-"* to specify **STDIN** as the input file.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

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.

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.

Security

The user of the **startrpnode** command needs write permission for the **IBM.PeerNode** resource class on each node that is to be started in the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Restrictions

This command must be run from a node that is online to the peer domain. The node that is to be brought online must be offline to the peer domain, must not be online to any other peer domain, and must be reachable from where the command is run.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/startrpnode

Examples

In this example, **nodeA** is defined and online to **AppIDomain**, **nodeB** is reachable from **nodeA**, and **nodeB** is not online to **AppIDomain** or any other peer domain. To bring **nodeB** online to **AppIDomain**, run this command from **nodeA**:

```
startrpnode nodeB
```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **forcerpoffline**, **lsrpnode**, **preprpnode**, **rmrpnode**, **stoprpnode**

stoprpdomain

Purpose

Takes an online peer domain offline.

Syntax

stoprpdomain [**–f**] [**–h]** [**–TV**] *domain_name*

Description

The **stoprpdomain** command takes all of the nodes that are currently online in the peer domain offline. The peer domain definition is not removed from the nodes.

The command must be run on a node that is online in the peer domain. If the command is run on a node that is offline to the peer domain, no action is performed.

The **-f** flag must be used to override a subsystem's rejection of the request to take the peer domain offline. A subsystem may reject the request if a peer domain resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystem that the peer domain must be brought offline regardless of the resource state.

Parameters

domain_name Specifies the name of the online peer domain that is to be brought offline.

Flags

-f Forces the subsystem to accept the stop request when it otherwise would not.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

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.

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.

Security

The user of the **stoprpdomain** command needs write permission for the **IBM.PeerDomain** resource class on each node that is defined to the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

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.

Location

/usr/sbin/rsct/bin/stoprpdomain

Examples

In these examples, **nodeA** is one of the nodes defined and is online to **ApplDomain**.

1. To take **ApplDomain** offline, run this command on **nodeA**:

   ```
   stoprpdomain ApplDomain
   ```

2. To take **ApplDomain** offline while making sure the stop request will not be rejected by any subsystem, run this command on **nodeA**:

   ```
   stoprpdomain -f ApplDomain
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **forcerpoffline**, **lsrpdomain**, **lsrpnode**, **mkrpdomain**, **preprpnode**, **startrpdomain**

stoprpnode

Purpose

Takes one or more nodes offline from a peer domain.

Syntax

stoprpnode [**–f**] [**–h**] [**–TV**] *node_name1* [*node_name2*...]

stoprpnode –F { *file_name* | *"–"* } [**–f**] [**–h**] [**–TV**]

Description

The **stoprpnode** command takes an online node offline from a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The **-f** flag must be used to override a subsystem's rejection of the request to take a node offline. A subsystem may reject the request if a node resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystems that the node must be brought offline regardless of the resource state.

If this command is used to take more than one node offline by specifying more than one *node_name* parameter, and the node that this command is running on is in the list, it will be brought offline last.

Parameters

node_name1 **[***node_name2***...]** Specifies the peer domain node names of the nodes that are to be brought offline from the peer domain. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Flags

-f Forces the subsystems to accept the stop request when it otherwise would not.

–F { *file_name* **|** *"–"* **}**
Reads a list of node names from *file_name*. Each line of the file is scanned for one node name. The pound sign (**#**) indicates that the remainder of the line (or the entire line if the **#** is in column 1) is a comment.

Use **-F** *"-"* to specify **STDIN** as the input file.

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

-T Writes the command's trace messages to standard error. For your software service organization's use only.

-V Writes the command's verbose messages to standard output.

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.

Standard input

When the **-F** ″**-**″ flag is specified, this command reads one or more node names from standard input.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. All verbose messages are written to standard output.

Standard error

All trace messages are written to standard error.

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.

Security

The user of the **stoprpnode** command needs write permission for the **IBM.PeerNode** resource class on each node that is to be stopped in the peer domain. By default, **root** on any node in the peer domain has read and write access to this resource class through the configuration resource manager.

Location

/usr/sbin/rsct/bin/stoprpnode

Restrictions

This command must be run on a node that is online to the peer domain. The node to be brought offline must be reachable from the node on which the command is run.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Examples

In these examples, **nodeA** and **nodeB** are online to **AppIDomain**.

1. To take **nodeB** offline, run this command on **nodeA**:

   ```
   stoprpnode nodeB
   ```

2. To take **nodeB** offline and force the offline request, run this command on **nodeA**:

   ```
   stoprpnode -f nodeB
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about peer domain operations

Commands: **addrpnode**, **lsrpnode**, **preprpnode**, **rmrpnode**, **startrpnode**

Part 3. Cluster configuration

Chapter 5. Configuration commands 145
ctadmingroup 146

Chapter 6. Configuration files 149
ctcas_hba2.map 150
ct_class_ids 152
ct_cssk.kf 153
ct_has.pkf 154
ct_has.qkf 156
ct_has.thl 158
ctcasd.cfg 160
ctrmc.acls 165
ctsec.cfg 166
ctsec_map.global, ctsec_map.local 167
unix.map 172

Chapter 7. Common Information Model (CIM) resource manager commands 175
lsassocmap 176
lsrsrcassoc 178
mkcimreg 181

© Copyright IBM Corp. 2002, 2007

Chapter 5. Configuration commands

© Copyright IBM Corp. 2002, 2007

ctadmingroup

Purpose

Defines a cluster administration group.

Syntax

To define a group:

ctadmingroup [**-h**] [**-TV**] *group_name*

To remove a group:

ctadmingroup -u [**-h**] [**-TV**] [*group_name*]

Description

The **ctadmingroup** command is used to define a cluster administration group. This command sets group ownership for trace files, so users who belong to a cluster administration group have the permissions needed to examine trace files that are produced by Reliable Scalable Cluster Technology (RSCT) subsystems. **ctadmingroup** changes existing trace files to the new permissions and group ownership. Trace files that are created after **ctadmingroup** is run will contain the new permissions. Note that this command does not create the specified group, nor does it add users to this group; it only gives users of this group access to the trace files.

If you run **ctadmingroup** with:

- a different group name, the new group that is specified becomes the cluster administration group, thereby replacing the previous group.
- no flags or parameters, it displays the group name and ID of the cluster administration group. If no cluster administration group is defined, this command does not produce any output.
- the **-u** flag, it removes the cluster administration group. After the group is removed, users who belong to that group may not be able to examine trace files. If no cluster administration group is defined, this command does not produce any output.

The location of the security subsystem's trace file is configurable. To determine the location of the trace file, **ctadmingroup** refers to the **/var/ct/cfg/ctcasd.cfg** file (if it is present) and the **/usr/sbin/rsct/cfg/ctcasd.cfg** file.

Parameters

group_name
: Specifies the name of the cluster administration group. This group must already exist in the group database (**/etc/group**, for example).

Flags

-u Removes the cluster administration group. After the group is removed, users who belong to that group may not be able to examine trace files. If no cluster administration group is defined, this command does not produce any output.

-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.

Files

/etc/group	The group database.
/var/ct/cfg/ctgroups	Stores the administration group name and caches the corresponding group ID.
/var/ct/cfg/ctcasd.cfg	The primary location of the cluster security configuration file, which contains the location of the security subsystem's trace file.
/usr/sbin/rsct/cfg/ctcasd.cfg	The secondary location of the cluster security configuration file. The **ctadmingroup** command refers to this file if the **/var/ct/cfg/ctcasd.cfg** file is not present.

Exit status

0 The command has run successfully.

1 The group name that was specified on the command line is not in the group database.

2 An internal error occurred.

3 An incorrect flag was entered on the command line.

4 An incorrect operand was entered on the command line.

Security

Only **root** users can run this command.

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.

Restrictions

Unpredictable results could occur if the mapping of the group name and group ID is changed after the command is run.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/ctadmingroup
Contains the **ctadmingroup** command

Examples

1. To display the group name and ID of the cluster administration group, enter:
   ```
   ctadmingroup
   ```

Related information

Files: **ctcasd.cfg**, **ctgroups**, **/etc/group**

Chapter 6. Configuration files

© Copyright IBM Corp. 2002, 2007

ctcas_hba2.map

Purpose

Defines the operating system identity that the RSCT enhanced host-based authentication (HBA2) security mechanism uses for service provider applications on a node.

Description

Applications that use the cluster security services library must obtain an identity from the security mechanisms supported by the library. These identities are specific to the individual security mechanisms supported by cluster security services. Because cluster security services supports multiple security mechanisms and multiple applications, the cluster security services library must be informed of which identity to use for an application when interacting with a specific security mechanism on its behalf.

The **ctcas_hba2.map** file defines the identities that the core cluster applications use when they interact with RSCT HBA2. The cluster security services library expects to find this file in **/var/ct/cfg/ctcas_hba2.map** (preferred) or **/usr/sbin/rsct/cfg/ctcas_hba2.map** (default).

This file is ASCII-text formatted, and can be modified with a standard text editor. However, this file should not be modified unless the administrator is instructed to do so by the cluster software service provider. If this configuration file is to be modified, the default **/usr/sbin/rsct/cfg/ctcas_hba2.map** file should not be modified directly. Instead, the file should be copied to **/var/ct/cfg/ctcas_hba2.map**, and modifications should be made to this copy. The default configuration file should never be modified.

All entries within this file use the following format:

```
SERVICE:service_name:user_name_running_the_service
```

Attribute	Definition
SERVICE	Required keyword
service_name	Specifies the name commonly used to refer to the application. For example, this could be the name used by the system resource controller to refer to this application.
user_name_running_the_service	Specifies the operating system user identity used to execute the application process. It is the owner identity that would be seen for the application process in the **ps** command output.

Files

/var/ct/cfg/ctcas_hba2.map

Restrictions

This file should not be modified unless the administrator is instructed to do so by the cluster software service provider. Incorrect modification of this file will result in authentication failures for the applications listed in this file and possibly their client applications. If this configuration file is to be modified, the default

/usr/sbin/rsct/cfg/ctcas_hba2.map file should not be modified directly. Instead, the file should be copied to **/var/ct/cfg/ctcas_hba2.map**, and modifications should be made to this copy. The default configuration file should never be modified.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/cfg/ctcas_hba2.map

Examples

This example shows the default contents of the configuration file:

```
SERVICE:ctrmc:root
SERVICE:rmc:root
SERVICE:ctloadl:loadl
SERVICE:ctdpcl:root
SERVICE:ctpmd:root
```

Related information

Commands: **ps**

Daemons: **ctcasd**

ct_class_ids

Purpose

Contains the mapping of resource class names to resource class IDs for the RMC subsystem.

Description

The **ct_class_ids** file contains the mapping of resource class names to resource class IDs for the RMC subsystem. This is a read-only file; the contents cannot be modified.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/cfg/ct_class_ids

ct_cssk.kf

Purpose

Contains the cluster shared secret key.

Description

In a peer domain with the cluster shared secret key (CSSK) function enabled, the configuration resource manager creates a file called **ct_cssk.kf** in the **/var/ct/***domain_name***/cfg** directory and stores the initial CSSK in it. A **ct_cssk.kf** file is then created on each node that is online in the peer domain.

The topology services subsystem uses the CSSK to provide message authentication, which ensures the integrity of messages that are sent between nodes within the peer domain. Once the CSSK function is enabled for a peer domain, all RMC, topology services, and group services message traffic is signed for authentication using the CSSK.

Any changes to the CSSK are coordinated across all nodes that are online in the peer domain and any offline nodes when they join the peer domain. The new key is distributed to all online nodes in the peer domain using the current CSSK. On each online node, the configuration resource manager replaces the key value in **/var/ct/***domain_name***/cfg/ct_cssk.kf** with the new value, and then refreshes the topology services subsystem to pick up the new key. Once refreshed, the new key is in effect for message authentication.

Security

The permissions of this file are **000**. Effectively, only **root** has read and write access to this file.

Restrictions

The configuration resource manager manages this file automatically. It should not be modified by any other user or program.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/var/ct/*domain_name***/cfg/ct_cssk.kf**

Related information

Commands: **mkrpdomain**, **rmrpdomain**, **rmrpnode**, **startrpdomain**, **startrpnode**

ct_has.pkf

Purpose

Default location for the local node's cluster security services public key file.

Description

The **/var/ct/cfg/ct_has.pkf** file is the default location where the **ctcasd** daemon expects to find the local node's public key file. The public key is stored in a proprietary binary format.

The **ctcasd.cfg** file permits the system administrator to specify an alternate location for this file. The **ctskeygen -p** command permits the administrator to create this file in an alternate location. If an alternate location is used, the file must meet all the criteria listed in the **Security** section of this man page. The file must not be recorded to a read-only file system, because this will prohibit the system administrator for modifying the contents of this file in the future.

If the **ctcasd** daemon cannot locate this file during its startup, it will check for the presence of the **ct_has.qkf** file. If both files are missing, the daemon assumes that it is being started for the first time after installation, and creates an initial private and public key file for the node. The daemon also creates the initial trusted host list file for this node. This file contains an entry for **localhost** and the host names and IP addresses associated with all of the active, IPv4- and IPv6-configured adapters that the daemon can detect. Inadvertent authentication failures could occur if the public and private key files were accidentally or intentionally removed from the local system before the daemon was restarted. **ctcasd** will create new keys for the node, which will not match the keys stored on the other cluster nodes. If RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) suddenly fails after a system restart, this is a possible source of the failure.

If the public key file is missing but the private key file is detected, the daemon concludes that the local node is misconfigured and terminates. A record is made to persistent storage to indicate the source of the failure.

Security

This file is readable to all users on the local system. Write permission is not granted to any system user.

By default, this file is stored in a locally-mounted file system. The **ctcasd.cfg** file permits system administrators to change the location of the file. Should system administrators use a different location, it is the administrator's responsibility to assure that the file is always accessible to the local node, and that all users from this local node can read the file. If the storage location does not meet these criteria, users and applications will be unable to authenticate to trusted services using RSCT HBA or HBA2.

If the system administrator chooses to place this file in a networked file system, the administrator must assure that no two nodes are attempting to use the same physical file as their own public key file. Because public keys differ between nodes, if two nodes attempt to use the same public key file, at least one of them will always obtain the incorrect value for its public key. This will cause applications and users from that node to fail authentication to trusted services within the cluster.

Restrictions

Cluster security services supports only its own private and public key formats and file formats. Secured Remote Shell formats are currently unsupported. Settings for the HBA_USING_SSH_KEYS attribute are ignored.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/var/ct/cfg/ct_has.pkf

Related information

Commands: **ctskeygen**

Daemons: **ctcasd**

Files: **ct_has.qkf**, **ct_has.thl**

ct_has.qkf

Purpose

Default location for the cluster security services private key file for the local node.

Description

The **/var/ct/cfg/ct_has.qkf** file is the default location where the **ctcasd** daemon expects to find the local node's private key file. The private key is stored in a proprietary binary format.

The **ctcasd.cfg** file permits the system administrator to specify an alternate location for this file. The **ctskeygen -q** command permits the administrator to create this file in an alternate location. If an alternate location is used, the file must meet all the criteria listed in the **Security** section of this man page. The file must not be recorded to a read-only file system, because this will prohibit the system administrator for modifying the contents of this file in the future

If the **ctcasd** daemon cannot locate this file during its startup, it will check for the presence of the **ct_has.pkf** file. If both files are missing, the daemon will assume that it is being started for the first time after installation, and create an initial private and public key file for the node. The daemon also creates the initial trusted host list file for this node. This file contains an entry for **localhost** and the host names and IP addresses associated with all of the active, IPv4- and IPv6-configured adapters that the daemon can detect. Inadvertent authentication failures could occur if the public and private key files were accidentally or intentionally removed from the local system before the daemon was restarted. **ctcasd** will create new keys for the node, which will not match the keys stored on the other cluster nodes. If RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) suddenly fails after a system restart, this is a possible source of the failure.

If the private key file is missing but the public key file is detected, the daemon concludes that the local node is misconfigured and terminates. A record is made to persistent storage to indicate the source of the failure.

Security

This file is readable and accessible only to the root user. Access to all other users is not provided.

By default, this file is stored in a locally mounted file system. The **ctcasd.cfg** file permits system administrators to change the location of the file. Should system administrators use a different location, it is the administrator's responsibility to assure that the file is always accessible to the local node, and that only the root user from this local node can access the file. If the storage location does not meet these criteria, the security of the node and the cluster should be considered compromised.

Restrictions

Cluster security services supports only its own private and public key formats and file formats. Secured Remote Shell formats are currently unsupported. Settings for the HBA_USING_SSH_KEYS attribute are ignored.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ct_has.qkf

Related information

Commands: **ctskeygen**

Daemons: **ctcasd**

Files: **ct_has.pkf**, **ct_has.thl**

ct_has.thl

Purpose

Default location for the local node's cluster security services trusted host list file.

Description

The **/var/ct/cfg/ct_has.thl** file is the default location where the **ctcasd** daemon expects to find the local node's trusted host list file. The contents of this file are stored in a proprietary binary format.

The trusted host list maps each host identity within the peer domain or management domain to the host's cluster security services public key. The **ctcasd** daemon uses this list to determine which nodes on the network are trusted, and to locate the public keys for these nodes to decrypt RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) credentials transmitted from another host within the cluster. If a host is not listed in a node's trusted host list, or if the public key recorded for that host is incorrect, the host will not be able to authenticate to that node using RSCT HBA or HBA2.

The **ctcasd.cfg** file permits the system administrator to specify an alternate location for this file. If an alternate location is used, the file must meet all the criteria listed in the **Security** section of this man page. The file must not be recorded to a read-only file system, because this will prohibit the system administrator for modifying the contents of this file in the future.

If the **ctcasd** daemon cannot locate this file during its startup, it will check for the presence of the **ct_has.pkf** file. If both files are missing, the daemon will assume that it is being started for the first time after installation, and create an initial private and public key file for the node. The daemon also creates the initial trusted host list file for this node. This file contains an entry for **localhost**, along with the IP addresses and the host names associated with all of the active, IPv4- and IPv6-configured adapters that the daemon can detect. Inadvertent authentication failures could occur if the public and private key files were accidentally or intentionally removed from the local system before the daemon was restarted. **ctcasd** will create new keys for the node, which will not match the keys stored on the other cluster nodes. If RSCT HBA or HBA2 suddenly fails after a system restart, this is a possible source of the failure.

Security

This file is readable by all users on the local system. Write access is not provided to any system user.

By default, this file is stored in a locally-mounted file system. The **ctcasd.cfg** file permits system administrators to change the location of the file. If the system administrator uses a different location, it is the administrator's responsibility to make sure the file is always accessible to the local node, and that all users from this local node can access the file. If the storage location does not meet these criteria, users and applications will be unable to authenticate to trusted services using RSCT HBA or HBA2.

If the system administrator chooses to place this file in a networked file system, the administrator must assure that no two nodes are attempting to use the same physical file as their own trusted host list file, or that the file does not contain an entry for localhost. By default, the trusted host list contains an entry for **localhost**,

which maps the local system's public key to this value. If multiple hosts share the same trusted host list file, attempts by users or applications to contact localhost for trusted services may fail because the entry maps to an incorrect public key value.

Restrictions

- Cluster security services supports only its own private and public key formats and file formats.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ct_has.thl

Related information

Commands: **ctkeygen**, **ctsthl**

Daemons: **ctcasd**

Files: **ct_has.pkf**, **ct_has.qkf**

ctcasd.cfg

Purpose

Provides operational parameters to the cluster security services daemon **ctcasd**.

Description

The **ctcasd.cfg** configuration file defines the operational parameters to the cluster security services daemon **ctcasd**. The **ctcasd** daemon reads this file when it (the daemon) initializes. The **ctcasd** daemon expects to find this configuration file in either the **/var/ct/cfg** directory (preferred) or in the **/usr/sbin/rsct/cfg** directory (default). System administrators can modify the contents of the file stored in the **/var/ct/cfg** directory, but should not modify the default version of the file in **/usr/sbin/rsct/cfg** unless instructed to do so by the cluster software service provider.

This file is ASCII-formatted, and can be modified using any available text editor. One attribute can be defined per line within this file. Attributes are specified as follows:

attribute=value

The following attributes are defined:

Attribute **Definition**

TRACE Indicates whether daemon tracing is activated. Valid values are: **ON** and **OFF**. If this attribute is *not* listed in the **ctcasd.cfg** file, tracing is activated by default (**TRACE=ON**). For coexistence with earlier versions of RSCT, **TRACE=false** is interpreted as **TRACE=OFF**.

TRACEFILE Specifies the fully-qualified path name where daemon tracing information is to be recorded. If this attribute is *not* listed in the **ctcasd.cfg** file, tracing information is recorded in **/var/ct/IW/log/ctsec/ctcasd/trace**.

TRACELEVELS

Indicates the tracing granularity employed by the daemon when tracing is activated. The possible trace categories are:

_SEC:Errors

Captures error information in the trace log. Valid values are: **1**, **2**, **4**, and **8**.

_SEC:Info

Traces the general execution progress of the daemon. Valid values are: **0**, **1**, **4**, and **8**.

The default value for this attribute is:
TRACELEVELS=_SEC:Info=1,_SEC:Errors=1 When setting the values of these trace categories, keep in mind that the lower the number is, the less intrusive (and less detailed) the trace will be. Multiple traces can be enabled at once. For example, if an administrator wants to enable a trace that captures basic execution information and highly-detailed error information, the specification for TRACELEVELS would be:
TRACELEVELS=_SEC:Info=1,_SEC:Errors=8

TRACESIZE Specifies the size of the trace file in bytes. The default value is 1 megabyte.

RQUEUESIZE Indicates the maximum length permitted for the daemon's internal run queue. If this value is not set, a default value of 64 is used.

MAXTHREADS
The limit to the number of working threads that the daemon may create and use at any given time (the ″high water mark″). If this value is not set, a default value of 10 is used.

MINTHREADS
The number of idle threads that the daemon will retain if the daemon is awaiting further work (the ″low water mark″). If this value is not set, a default value of 4 is used.

THREADSTACK
Sets the internal memory used by the daemon for thread stack space. The value is expressed in bytes. If no value is specified, the default system thread stack size is used. This value should not be modified by the administrator unless instructed to do so by IBM Service.

HBA_USING_SSH_KEYS
Indicates whether the daemon is making use of Secured Remote Shell keys. Acceptable values are **true** and **false**. If this value is not defined, a default value of **false** is used. See **Restrictions**.

HBA_PRVKEYFILE
Provides the full path name of the file that contains the local node's private key. If this value is not set, the default location of **/var/ct/cfg/ct_has.qkf** is used.

HBA_PUBKEYFILE
Provides the full path name of the file that contains the local node's public key. If this value is not set, the default location of **/var/ct/cfg/ct_has.pkf** is used.

HBA_THLFILE
Provides the full path name of the file that contains the local node's trusted host list. If this value is not set, the default location of **/var/ct/cfg/ct_has.thl** is used.

HBA_KEYGEN_METHOD
Indicates the method to be used by **ctcasd** to generate the private and public keys of the local node if the files containing these keys do not exist. Acceptable values are those that can be provided as arguments to the **ctskeygen -m** command. If no value is provided for this attribute, the default value of **rsa512** is used.

HBA_CRED_TIMETOLIVE
Sets the life span of RSCT host-based authentication (HBA) credentials (credentials created and verified using the **unix** MPM mnemonic). The credential life span dictates the period of time after a credential is created that the RSCT HBA mechanism should consider the credential valid. Setting a credential life span enables the RSCT HBA mechanism to detect outdated credentials and refuse authentication to applications presenting such credentials. If no value is specified for this keyword (the default), credentials will not be checked for expiration.

For more information on using this keyword, see the *RSCT: Administration Guide*.

HBA2_CRED_CTX_LIFETIME
Sets the expiration time for a security context that is established using the RSCT enhanced host-based authentication (HBA2) mechanism. Once the security context is established, the context will remain valid for the length of time specified by this parameter. After this amount of time passes, the client and server applications will need to re-establish the security context.

If no value is specified for this parameter, the RSCT HBA2 MPM will use a default value of 43 200 seconds (12 hours). The default value is -1, indicating that security contexts established using the RSCT HBA2 MPM will not expire.

HBA2_CRED_TIMETOLIVE
Sets the life span of RSCT HBA2 credentials (credentials created and verified using the **hba2** MPM mnemonic). The credential life span dictates the period of time after a credential is created that the RSCT HBA2 mechanism should consider the credential valid. Setting a credential life span enables the RSCT HBA2 mechanism to detect outdated credentials and refuse authentication to applications presenting such credentials.

If no value is specified for this keyword, credential tracking is not performed and credentials will not be checked for expiration. The default value is 300 seconds (5 minutes).

For more information on using this keyword, see the *RSCT: Administration Guide*.

HBA2_NONCE_FILEMIN
Indicates the minimum number of credential identities retained by the RSCT HBA2 mechanism between executions of the **ctcasd** daemon. Whenever the RSCT HBA2 MPM authenticates a credential, the identity information for that credential is stored and used in subsequent authentication attempts to detect repeat uses of the same credential. The **ctcasd** daemon creates a file and reserves enough file system space so that the RSCT HBA2 MPM can store the minimum number of credential identities. When the **ctcasd** daemon starts, it reads the contents of this file into memory and uses it in subsequent authentication checks using the RSCT HBA2 MPM. This permits **ctcasd** and the RSCT HBA2 MPM to check for re-used credentials from prior executions of the daemon if the **ctcasd** daemon has been shut down.

If no value is specified for this parameter, the **ctcasd** daemon uses a default value of 4096.

SERVICES Lists the internal cluster security services library services that the daemon supports. This entry should not be modified by system administrators unless they are explicitly instructed to do so by the cluster security software service provider.

Files

/usr/sbin/rsct/cfg/ctcasd.cfg	Default location of the **ctcasd.cfg** file
/var/ct/cfg/ct_has.pkf	Default location of the local node's public key
/var/ct/cfg/ct_has.qkf	Default location of the local node's private key
/var/ct/cfg/ct_has.thl	Default location of the local node's trusted host list

/var/ct/IW/log/ctsec/ctcasd/trace
Default location where tracing information is recorded

Restrictions

Cluster security services supports only its own private and public key formats and file formats. Secured Remote Shell formats are currently unsupported. Settings for the HBA_USING_SSH_KEYS attribute are ignored.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/var/ct/cfg/ctcasd.cfg

Examples

This example shows the default contents of the configuration file:

```
TRACE= ON
TRACEFILE= /var/ct/IW/log/ctsec/ctcasd/trace
TRACELEVELS= _SEC:Info=1,_SEC:Errors=1
TRACESIZE= 1003520
RQUEUESIZE=
MAXTHREADS=
MINTHREADS=
THREADSTACK= 131072
HBA_USING_SSH_KEYS= false
HBA_PRVKEYFILE=
HBA_PUBKEYFILE=
HBA_THLFILE=
HBA_KEYGEN_METHOD= rsa512
HBA_CRED_TIMETOLIVE=
HBA2_CRED_CTX_LIFETIME= -1
HBA2_CRED_TIMETOLIVE= 300
HBA2_NONCE_FILEMIN= w
SERVICES= hba CAS
```

After modification, the contents of the configuration file might look like this:

```
TRACE=ON
TRACEFILE=/var/ct/IW/log/ctsec/ctcasd/trace
TRACELEVELS=SEC:Info=1,_SEC:Errors=8
TRACESIZE=1003520
RQUEUESIZE=64
MAXTHREADS=10
MINTHREADS=4
THREADSTACK=131072
HBA_USING_SSH_KEYS= false
HBA_PVTKEYFILE=/var/ct/cfg/qkey
HBA_PUBKEYFILE=/var/ct/cfg/pkey
HBA_THLFILE=/var/ct/cfg/thl
HBA_KEYGEN_METHOD=rsa512
HBA_CRED_TIMETOLIVE=90s
HBA2_CRED_CTX_LIFETIME= -1
HBA2_CRED_TIMETOLIVE= 300
HBA2_NONCE_FILEMIN= w
SERVICES=hba CAS
```

Related information

Commands: **ctadmingroup**, **ctskeygen**, **date**

Daemons: **ctcasd**

Files: **ct_has.pkf**, **ct_has.qkf**, **ct_has.thl**

ctrmc.acls

Purpose

Contains a node's resource monitoring and control (RMC) access control list (ACL).

Description

RMC implements authorization using an access control list (ACL) file. Specifically, RMC uses the ACL file on a particular node to determine the permissions that a user must have in order to access resource classes and their resource instances. A node's RMC ACL file is named **ctrmc.acls** and is installed in the **/usr/sbin/rsct/cfg** directory. You can allow RMC to use the default permissions set in this file, or you can use the **chrmcacl** command to modify these defaults. See "chrmcacl " on page 4 for more information.

For more information about the RMC ACL file, see the *RSCT: Administration Guide*.

For information about how access controls are implemented for the **IBM.LPCommands** resource class and its resources, see "lpacl " on page 422.

Files

/usr/sbin/rsct/cfg/ctrmc.acls	Default location of the **ctrmc.acls** file
/var/ct/IW/log/mc/default	Location of any errors found in the modified **ctrmc.acls** file

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/var/ct/cfg/ctrmc.acls

Related information

"lpacl " on page 422

Books: *RSCT: Administration Guide*

Commands: **chrmcacl**

ctsec.cfg

Purpose

Provides configuration information about the authentication methods that cluster security services can use for client/server authentication.

Description

The **ctsec.cfg** configuration file provides configuration information about the authentication methods that cluster security services can use for client-server authentication. Each authentication method is handled by a mechanism pluggable module (MPM). Each MPM's configuration is defined by a one-line entry in the **ctsec.cfg** file. The entry contains information about:

- the priority of the MPM when cluster security services chooses the authentication method for the client-server authentication
- the numeric code of the MPM, which is unique among all of the MPMs in the configuration file
- the mnemonic of the MPM, which is unique among all of the MPMs in the configuration file
- the name of the binary module that implements the functionality of the MPM
- miscellaneous flags used by cluster security services' mechanism abstract layer (MAL) when handling the MPM

Cluster security services include a default **ctsec.cfg** file in the **/usr/sbin/rsct/cfg/** directory. Use the **ctscfg** command to modify a working copy of this configuration file. **ctscfg** does not modify the default configuration file in in **/usr/sbin/rsct/cfg/**. Instead, **ctscfg** makes a copy (if one does not exist already) of the default **ctsec.cfg** file and copies it to the **/var/ct/cfg/** directory. If a working copy of this file does exist already and there is enough space, the previous version is recorded to **/var/ct/cfg/ctsec.cfg.bak**.

Files

/var/ct/cfg/ctsec.cfg	Working copy of the MAL's configuration file
/var/ct/cfg/ctsec.cfg.bak	Backup of the working copy of the MAL's configuration file

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/cfg/ctsec.cfg

Related information

Commands: **ctscfg**

ctsec_map.global, ctsec_map.local

Purpose

Associates operating system user identifiers on the local system with network security identifiers for authorization purposes.

Description

RSCT trusted services use the identity mapping definition files **ctsec_map.global** and **ctsec_map.local** to determine whether an RSCT client application's user should be granted access to specific RSCT functions and resources. These files are used to associate security network identifiers that are used by RSCT's cluster security services with user identifiers on the local system. RSCT trusted services use these files to determine what association, if any, exists for the RSCT client, and then use this association while examining the RSCT access controls to determine whether the RSCT client should be granted access.

Two identity mapping definition files can be used:

- The **ctsec_map.global** file contains associations that are to be recognized on all nodes within the cluster configuration
- The **ctsec_map.local** file contains associations that are specific to a particular node

In a cluster configuration, all **ctsec_map.global** files should be the same. Any local system additions that are required for that specific system should be made in the **ctsec_map.local** file.

RSCT provides a default **ctsec_map.global** file in the **/usr/sbin/rsct/cfg** directory. Do *not* change this file. If you need to add more associations for the cluster, copy this file to the **/var/ct/cfg** directory. Make any changes to this new file: **/var/ct/cfg/ctsec_map.global**. Any entries that exist in the default **ctsec_map.global** file must exist in the replacement version of the file in the **/var/ct/cfg** directory, or the RSCT trusted services may refuse access to other RSCT trusted services peers. RSCT does not provide a default **ctsec_map.local** file. The administrator can create this file, which must reside in the **/var/ct/cfg** directory as well.

ctsec_map.global and **ctsec_map.local** are ASCII-formatted files that can be viewed and modified using a text editor. Each line in the file constitutes an entry. Blank lines and lines that start with a pound sign (**#**) are ignored. Each entry is used to either associate a security network identifier with a local operating system user identifier, or to expressly state that no association is allowed for a security network identifier.

Ordering of entries within these files is important. Cluster security services parses the **ctsec_map.global**and **ctsec_map.local** files as follows:

1. If the **/var/ct/cfg/ctsec_map.local** file exists, cluster security services checks for associations in this file
2. If the **/var/ct/cfg/ctsec_map.global** file exists, cluster security services checks for associations in this file
3. Otherwise, cluster security services checks for associations within the **/usr/sbin/rsct/cfg/ctsec_map.global**, if this file exists

The first entry that successfully grants or denies an association for a security network identifier in this search path is the one that cluster security services uses. If

entries in both the **ctsec_map.global**and **ctsec_map.local** files grant differing associations to the same security network identifier, cluster security services will use the association stated by the entry in the **ctsec_map.local** file. Also, if two entries within the **ctsec_map.global** file grant different associations to the same security network identifier, cluster security services will use the association granted by the entry listed earlier in the **ctsec_map.global** file. You can use the **ctsidmck** command to verify the association rule that is used by cluster security services for specific security network identifiers.

Cluster security services recognizes these characters as reserved: **<**, **>**, **:**, **=**, **!**, **@**, *****, and considers these, along with white space characters, as token separators. The wildcard character ***** is permitted, but should not be used multiple times between other token separator characters. Contents of the identity mapping definition files use the following Backus-Nour format:

```
<mapping_entry> ::= <mechanism_mnemonic> ':' <mapping>>

<mechanism_mnemonic> ::= 'unix', 'krb5'

<mapping> ::= <explicit mapping> | <mapping_rule>

<explicit_mapping> ::= <source_mapping> '=' <local_user_identity>
                       | '!' <source_mapping>

<source_mapping> ::= <network_identity> | <match_pattern>'*'

<target_mapping> ::= <mapped_identity> | '*'

<network_identity> ::= <user_name>'@'<registry_name>

<user_name> ::= <match_pattern>'*' | '*'

<registry_name> ::= <match_pattern> | '*' | <mpm_defined_reserved_word>

<mpm_defined_reserved_word> ::= '<'<alphanumeric_string>'>'

<mapped_identity> ::= <alphanumeric_string>

<match_pattern> ::= null string | <alphanumeric_string>

<alphanumeric_string> ::= any non-empty array of alphanumeric characters not
                          consisting of the reserved token separator characters
```

An <mpm_defined_reserved_word> is a special instruction to the underlying security mechanism associated with the security network identifier that instructs the mechanism to interpret the identifier in a specific manner. The following reserved words are defined:

<iw> A reserved word for security network identities using the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism. This keyword maps the RSCT HBA or HBA2 **root** network identity of the local node to the **root** user. When the cluster security services identity mapping program processes the **ctsec_map.global** file, it replaces the **<iw>** keyword with the node ID of the node.

<cluster>
A reserved word for security network identities using the RSCT HBA or HBA2 security mechanism. The mapping entry is applied to a security network identifier if the identifier is known to originate from any host within the cluster that is currently active for the local node.

<any_cluster>
A reserved word for security network identities using the RSCT HBA or HBA2 security mechanism. The mapping entry is applied to a security network identifier if the identifier is known to originate from any host within any cluster that the local node is currently defined. The local node does not need to be active within that cluster when the mapping is applied.

<realm>
A reserved word for security network identities using the Kerberos version 5 mechanism. The mapping entry is applied to a security network identity if the identifier is known to originate within the Kerberos realm that is currently active. See **Restrictions.**

Security

- The default identity mapping definition file **/usr/sbin/rsct/cfg/ctsec_map.global** is readable by all system users, but permissions prevent this file from being modified by any system user.
- When creating the override identity mapping definition files **/var/ct/cfg/ctsec_map.global** and **/var/ct/cfg/ctsec_map.local**, make sure that the files can be read by any system user, but that they can only be modified by the root user or other restrictive user identity not granted to normal system users.
- By default, these files reside in locally-mounted file systems. While it is possible to mount the **/var/ct/cfg** directory on a networked file system, this practice is discouraged. If the **/var/ct/cfg/ctsec_map.local** file were to reside in a networked file system, any node with access to that networked directory would assume that these definitions were specific to that node alone when in reality they would be shared.

Restrictions

RSCT does not support the Kerberos version 5 mechanism. Any entries using the mechanism mnemonic **krb5** or the reserved word **<realm>** will not be applied.

Implementation specifics

These files are part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. The default file is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/cfg/ctsec_map.global
Contains the default identity mapping definition file.

/var/ct/cfg/ctsec_map.global
Contains the replacement for the default global identity mapping definition file. Any entries that exist in the default **ctsec_map.global** file must be replicated in this file, or necessary access required by RSCT trusted services clients will be refused. This file contains identity mapping definitions expected to be recognized by all nodes within the cluster. It is expected that this file will have the same contents for each node within the cluster.

/var/ct/cfg/ctsec_map.local
Contains additional identity mapping definitions specific to the local node. This file adds identity mapping definitions to the set recognized for the entire cluster. Entries within this file are applied before entries from the **ctsec_map.global** file. It is expected that

the contents of this file will vary from node to node within the cluster, and provide mappings required for clients that access the local node only.

Examples

These reserved characters: **<**, **>**, **:**, **=**, **!**, and **@**, are interpreted as token separators, as are white space characters.

Examples of valid identity mapping definition entries:

unix:zathras@epsilon3.ibm.com=zathras
This entry grants the association for the RSCT HBA or HBA2 security mechanism identity **zathras@epsilon3.ibm.com** to the local user identifier **zathras**. This entry will not be applied to other RSCT HBA or HBA2 identities.

unix:!zathras@greatmachine.net
This entry denies any local user identity association for the RSCT HBA or HBA2 identity **zathras@greatmachine.net**. This entry will not be applied to other RSCT HBA or HBA2 identities.

unix:entilzah@<cluster>=root
The <cluster> reserved word will match any RSCT HBA or HBA2 identity containing the user name **entilzah** that originates from any host within the currently-active cluster. This will grant associations for such RSCT HBA or HBA2 identities as **entilzah@anglashok.ibm.com** and **entilzah@mimbar.ibm.com** to the local user **root** when the local node is active within the cluster that also contains the hosts **anglashok.ibm.com** and **mimbar.ibm.com**. Associations will not be granted for such RSCT HBA or HBA2 identities as **entilzah@whitestar.ibm.com** if the host **whitestar.ibm.com** is not part of the cluster that is currently active.

unix:entilzah@<any_cluster>=root
The <cluster> reserved word will match any RSCT HBA or HBA2 identity containing the user name **entilzah** that originates from any host within the currently-active cluster. This will grant associations for RSCT HBA or HBA2 identities such as **entilzah@anglashok.ibm.com** and **entilzah@mimbar.ibm.com** to the local user **root** when the local node is active within the cluster that also contains the hosts **anglashok.ibm.com** and **mimbar.ibm.com**. Associations will also be granted for RSCT HBA or HBA2 identities such as **entilzah@whitestar.ibm.com** to the local user **root** if the host **whitestar.ibm.com** is part of any cluster known to the local host.

unix:zathras@*=zathras The ***** character in this entry will match any RSCT HBA or HBA2 identity that contains the user name **zathras** from any host to the local user identifier **zathras**. This will grant associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** and **zathras@greatmachine.net** to the local user identifier **zathras**.

unix:zathras@*.ibm.com=zathras
The ***** character in this entry will match any RSCT HBA or HBA2 identity that contains the user name **zathras** and a host name ending with an **ibm.com** network domain to the local user identifier **zathras**. This will grant associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** and **zathras@newibm.com** to the local user identifier **zathras**.

unix:*@epsilon3.ibm.com=zathras
The * character in this entry will match any RSCT HBA or HBA2 identity from the host **epsilon3.ibm.com** and associate that client to the local user **zathras**. This will grant associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** and **draal@epsilon3.ibm.com** to the local user identifier **zathras**.

unix:*@epsilon3.ibm.com=* The * characters in this entry will match any RSCT HBA or HBA2 identity from the host **epsilon3.ibm.com** and associate that client to the local user whose name matches the user name from the security network identifier. This will grant associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** to the local user **zathras** and **draal@epsilon3.ibm.com** to the local user identifier **draal**.

unix:!*@epsilon3.ibm.com The * characters in this entry will match any RSCT HBA or HBA2 identity from the host **epsilon3.ibm.com** and deny any association for that client to any local user. This will deny associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** and **draal@epsilon3.ibm.com**, but will not deny associations for the UNIX HBA network identifier **zathras@greatmachine.net**.

unix:*@*=* The * characters in this entry will match any RSCT HBA or HBA2 identity from any host and associate that client to the local user whose name matches the user name from the security network identifier. This will grant associations for RSCT HBA or HBA2 identities such as **zathras@epsilon3.ibm.com** to the local user **zathras** and **entilzah@anglashok.ibm.com** to the local user identifier **entilzah**.

Examples of identity mapping definition entries that are *not* valid:

***:zathras@epsilon3.ibm.com=zathras**
The security mechanism cannot be determined. Each entry must explicitly name a security mechanism that needs to be applied to interpret the entry.

unix:zathras@epsilon3.ibm.com=z*
The local user identity to use is ambiguous.

unix:zathras@*.ibm.*=zathras
This entry repeats wildcard characters between the token separators **@** and **=**, which makes the entry ambiguous.

unix:*athra*@epsilon3.ibm.com=zathras
This entry repeats wildcard characters between the token separators **:** and **@**, which makes the entry ambiguous.

unix:*=* The wildcard character * is ambiguous. It cannot be determined if the wildcard character applies to the identity name or the identity location.

Related information

Commands: **ctsidmck**

unix.map

Purpose

Defines the operating system identity that the RSCT host-based authentication (HBA) security mechanism uses for service provider applications on a node.

Description

Applications that use the cluster security services library must obtain an identity from the security mechanisms supported by the library. These identities are specific to the individual security mechanisms supported by cluster security services. Because cluster security services supports multiple security mechanisms and multiple applications, the cluster security services library must be informed of which identity to use for an application when interacting with a specific security mechanism on its behalf.

RSCT HBA is the default security mechanism that the cluster security services library uses. The **unix.map** file defines the identities that the core cluster applications use when they interact with RSCT HBA. The cluster security services library expects to find this file in **/var/ct/cfg/unix.map** (preferred) or **/usr/sbin/rsct/cfg/unix.map** (default).

This file is ASCII-text formatted, and can be modified with a standard text editor. However, this file should not be modified unless the administrator is instructed to do so by the cluster software service provider. If this configuration file is to be modified, the default **/usr/sbin/rsct/cfg/unix.map** file should not be modified directly. Instead, the file should be copied to **/var/ct/cfg/unix.map**, and modifications should be made to this copy. The default configuration file should never be modified.

All entries within this file use the following format:

```
SERVICE:service_name:user_name_running_the_service
```

Attribute	Definition
SERVICE	Required keyword
service_name	Specifies the name commonly used to refer to the application. For example, this could be the name used by the system resource controller to refer to this application.
user_name_running_the_service	Specifies the operating system user identity used to execute the application process. It is the owner identity that would be seen for the application process in the **ps** command output.

Files

/var/ct/cfg/unix.map

Restrictions

This file should not be modified unless the administrator is instructed to do so by the cluster software service provider. Incorrect modification of this file will result in authentication failures for the applications listed in this file and possibly their client applications. If this configuration file is to be modified, the default **/usr/sbin/rsct/cfg/unix.map** file should not be modified directly. Instead, the file

should be copied to **/var/ct/cfg/unix.map**, and modifications should be made to this copy. The default configuration file should never be modified.

Implementation specifics

This file is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/cfg/unix.map

Examples

This example shows the default contents of the configuration file:

```
SERVICE:ctrmc:root
SERVICE:rmc:root
SERVICE:ctloadl:loadl
SERVICE:ctdpcl:root
SERVICE:ctpmd:root
```

Related information

Commands: **ps**

Daemons: **ctcasd**

Chapter 7. Common Information Model (CIM) resource manager commands

© Copyright IBM Corp. 2002, 2007

lsassocmap

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 have been registered with the CIM resource manager are displayed in table format, similar to the output of the **lscondresp** command.

If you specify **lsassocmap** without any parameters, all association classes, endpoints, and ″roles″ are displayed. A *role* is the name of the class's 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 ” on page 178 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 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'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.

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

When the **-T** flag is specified, this command's trace messages are written to standard error.

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 specified association class could not be found.

Implementation specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack.

Location

/usr/sbin/rsct/bin/lsassocmap

Examples

To display associations that are available in a cluster, enter:

```
lsassocmap
```

The output will look like this:

```
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
```

Related information

Commands: **lsrsrcassoc**, **lscondresp**, **mkcimreg**

lsrsrcassoc

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 have been registered with the CIM resource manager are called to retrieve association data. Before using **lsrsrcassoc**, it may 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's output is similar to that of **lsrsrc**. Resources associated with a given 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 those that are 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 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'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.

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

When the **-T** flag is specified, this command's trace messages are written to standard error.

Exit status

0 The command has run 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.

Location

/usr/sbin/rsct/bin/lsrsrcassoc

Examples

To view instances of **cimv2.IBMAIX_UnixProcess** that are associated with **cimv2.IBMAIX_OperatingSystem** on the specified node, enter:

```
lsrsrcassoc -c cimv2.IBMAIX_OSProcess -s 'Name=~"c175nf14"' -S \
'Name=~"emacs"' cimv2.IBMAIX_OperatingSystem Handle Parameters
```

In this example:

- **-c cimv2.IBMAIX_OSProcess** is the association class whose provider is used.
- **-s 'Name=~"c175nf14"'** is the selection string against the **cimv2.IBMAIX_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** objects; only those with **Name** attributes that contain the pattern **emacs** are returned.
- **cimv2.IBMAIX_OperatingSystem**, which is the "source object" parameter, is 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 output will look like this:

```
Resource Persistent Attributes for cimv2.IBMAIX_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"}
```

Related information

Commands: **lsassocmap**, **lsrsrc**, **mkcimreg**

mkcimreg

Purpose

Registers CIM classes and CMPI providers with RMC.

Syntax

To register a class:

mkcimreg [**–l** *include_directory...*] [**–f**] [**–h**] *definition_file...*

To register a provider:

mkcimreg [**–l** *include_directory...*] [**–p** *provider_directory*] [**–h**] *registration_file...*

To compile the CIM schema:

mkcimreg [**–l** *include_directory...*] **–b** *schema_path* [**–h**]

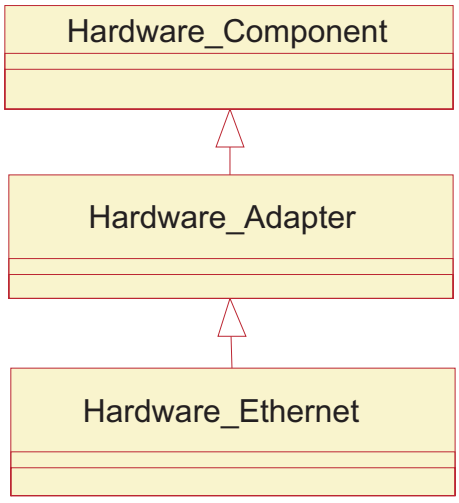
Description

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 **-l** 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

Use the **-f** flag to register a class that already exists in the current namespace. With this flag, any existing class registration data is overwritten with the definition that is provided in the class definition file. Without this flag, class registration is rejected in case the class that has been registered before is already on the system.

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:



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:

   ```
   /usr/sbin/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:

   ```
   /usr/sbin/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.

Parameters

definition_file...
Specifies one or more class definition files.

registration_file...
Specifies one or more provider registration files.

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.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

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.

Security

This command requires **root** authority.

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.

Location

/usr/sbin/rsct/bin/mkcimreg

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.

Related information

Books: *RSCT: Administration Guide*, for information about viewing instance property values

Commands: **lssrc**, **rmcctrl**, **stopsrc**

Files: **ct_class_ids**

Web sites: **http://www.dmtf.org**, for information about the Distributed Management Task Force (DMTF), which develops and maintains CIM

Part 4. Monitoring resources

Chapter 8. Event-response resource manager (ERRM) commands 187
chcondition 188
chresponse 193
lscondition 198
lscondresp 204
lsevent 210
lsresponse 215
mkcondition 221
mkcondresp 228
mkresponse 231
rmcondition 237
rmcondresp 240
rmresponse 244
startcondresp 247
stopcondresp 251

Chapter 9. ERRM scripts 255
elogevent, logevent 256
enotifyevent, notifyevent 258
ewallevent, wallevent 260
snmpevent 262

Chapter 10. Sensor resource manager commands 265
chsensor 266
lssensor 270
mksensor 276
refsensor 282
rmsensor 286

© Copyright IBM Corp. 2002, 2007

Chapter 8. Event-response resource manager (ERRM) commands

© Copyright IBM Corp. 2002, 2007

chcondition

Purpose

Changes the attributes of a defined condition.

Syntax

To change a condition's attributes:

chcondition [**–r** *resource_class*]
[**–e** *"event_expression"*] [**–E** *"rearm_expression"*]
[**–d** *"event_description"*] [**–D** *"rearm_description"*]
[**–m l** | **m** | **p**] [**–n** *node_name1*[**,***node_name2*...]]
[**––qnotoggle** | **––qtoggle**]
[**–s** *"selection_string"*] [**–S c** | **w** | **i**]
[**–h**] [**–TV**] *condition*[**:***node_name*]

To rename a condition:

chcondition –c *new_condition* [**–h**] [**–TV**] *condition*[**:***node_name*]

To lock or unlock a condition:

chcondition { **-L** | **-U** } [**–h**] [**–TV**] *condition*[**:***node_name*]

Description

The **chcondition** command changes the attributes of a defined condition. If you use the **-c** flag to change the name of the condition, any condition/response associations remain intact.

If a particular condition is needed for system software to work properly, it may be locked. A locked condition cannot be modified or removed until it is unlocked. If the condition you specify with the **chcondition** command is locked, it will not be modified; instead, an error is generated informing you that the condition is locked. To unlock a condition, you can use the **-U** flag. However, because a condition is typically locked so that system software works properly, you should exercise caution before unlocking it. To lock a condition, so that it cannot be modified, use the **-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.*

Parameters

condition Specifies the name of an existing condition that is defined on *node_name*.

node_name Specifies the node in a 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.

Flags

–c *new_condition*
Assigns a new name to the condition. *new_condition*, which replaces the current name, is a character string that identifies the condition. If *new_condition* contains one or more spaces, it must be enclosed in quotation marks. A name cannot be null, consist of all spaces, or contain embedded double quotation marks.

–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 a dynamic attribute of *resource_class*, a mathematical comparison symbol (**>**, for example), and a constant.

–L Locks a condition so it cannot be modified or removed. When locking a condition using the **-L** flag, no other operation can be performed by this command.

–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. The condition applies only to the local node (the node where the condition is defined). 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. 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. 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.

––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 which resource class this condition will monitor. The **lsrsrcdef** command can be used to list the resource class names.

–s *"selection_string"*
Specifies a selection string that is applied to all of the *resource_class* attributes to determine which resources *event_expression* should monitor. The default is to monitor all resources within *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)

–U Unlocks a condition so it can be modified or removed. If a condition is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition using the **-U** flag, no other operation can be performed by this command.

–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.

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'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.

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.

Security

The user of the **chcondition** command needs write permission to the **IBM.Condition** resource class on the node where the condition is defined. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chcondition

Examples

These examples apply to standalone systems:

1. To change the condition name from ″FileSystem space used″ to ″Watch FileSystem space″, run this command:

```
chcondition -c "Watch FileSystem space"  "FileSystem space used"
```

2. To change a rearm expression and rearm description for a condition with the name ″tmp space used″, run this command:

```
chcondition -E "PercentTotUsed < 80" \
-D "Start monitoring tmp again after it is less than 80 percent full" \
"tmp space used"
```

In the following examples, which apply to management domains, the node on which the command is run is on the management server.

1. To change the condition with the name ″FileSystem space used″ on the management server to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" "FileSystem space used"
```

2. To change the condition with the name ″NodeB FileSystem space used″ on **NodeB** to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" \
"NodeB FileSystem space used":NodeB
```

This example applies to a peer domain:

1. To change the condition defined on **NodeA** with the name ″FileSystem space used″ to check for space usage that is greater than 95%, run this command:

```
chcondition -e "PercentTotUsed > 95" \
"FileSystem space used":NodeA
```

Related information

“rmccli ” on page 74

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about ERRM operations and about how to use expressions and selection strings

Commands: **lscondition**, **lscondresp**, **mkcondition**, **nodegrp**, **rmcondition**

chresponse

Purpose

Adds or deletes the actions of a response or renames a response.

Syntax

To add an action to a response:

chresponse –a –n *action* [**–d** *days_of_week*[**,***days_of_week...*]] [**–t** *time_of_day*[**,***time_of_day...*]] [**–s** *action_script*] [**–r** *return_code*] [**–e a** | **r** | **b**] [**–o**] [**–E** *env_var*=*value*[**,***env_var*=*value...*]] [**–u**] [**–h**] [**–TV**] *response*[**:***node_name*]

To delete an action from a response:

chresponse –p –n *action* [**–h**] [**–TV**] *response*[**:***node_name*]

To rename a response:

chresponse –c *new_response* [**–h**] [**–TV**] *response*[**:***node_name*]

To lock or unlock a response:

chresponse { **-L** | **-U** } [**–h**] [**–TV**] *response*[**:***node_name*]

Description

The **chresponse** command adds an action to a response or deletes an action from a response. Actions define commands to be run when the response is used with a condition and the condition occurs. The **chresponse** command can also be used to rename a response.

If a particular response is needed for system software to work properly, it may be locked. A locked response cannot be modified or removed until it is unlocked. If the response you specify with the **chresponse** command is locked, it will not be modified; instead, an error is generated informing you that the response is locked. To unlock a response, you can use the **-U** flag. However, because a response is typically locked for system software to work properly, you should exercise caution before unlocking it. To lock a response, so that it cannot be modified, use the **-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.*

Parameters

response Specifies the name of the response to be changed.

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.

Flags

–a Adds the action specification to *response*.

–p Deletes *action* from *response*.

–c *new_response*
Specifies a new name to assign to the response. The new name must not already exist. The new name replaces the current name. The *new_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.

–n *action*
Specifies the name of the action. When the **–a** flag is used, this is the name of the action being defined. When the **–p** flag is used, this is the name of the action to be deleted. Action names must be unique within a response. Only one action can be defined at a time.

–d *days_of_week*[**,***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

–t *time_of_day*[**,***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 is **0000-2400**. If no value is specified but a comma is entered, the default value is used.

–s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the man pages for **displayevent**, **logevent**, **notifyevent**, and **wallevent** for descriptions of predefined response scripts that are provided with the application.

–r *return_code*
Specifies the expected return code for *action_script*. The actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating whether they match. If the **–r** flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

–e a | **r** | **b**
Specifies the type of event that causes the action being defined to run:

a Specifies an event. This is the default.

r Specifies a rearm event.

b Specifies both an event and a rearm event.

–o Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

–E *env_var*=*value*[,*env_var*=*value...*]
Specifies any environment variables to be set before *action_script* is run. If multiple *env_var*=*value* variables are specified, they must be separated by commas.

–u Specifies that the action is to be run when a monitored resource becomes undefined.

–L Locks a response so it cannot be modified or removed. When locking a response using the **-L** flag, no other operation can be performed by this command.

–U Unlocks a response so it can be modified or removed. If a response is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a response using the **-U** flag, no other operation can be performed by this command.

–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.

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'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.

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.

Security

The user of the **chresponse** command needs write permission to the **IBM.EventResponse** resource class on the node where the response is defined. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chresponse

Examples

These examples apply to standalone systems:

1. In this example, the action named ″E-mail root″ cannot be the only action. To delete ″E-mail root″ from the response named ″E-mail root anytime″, run this command:

```
chresponse -p -n "E-mail root" "E-mail root anytime"
```

2. In this example, the action named ″E-mail root″ will be used Monday through Friday from 8 AM to 6 PM, will use the command **/usr/sbin/rsct/bin/notifyevent root**, will save standard output in the audit log, and will expect return code 5 from the action. To add ″E-mail root″ to the response named ″E-mail root anytime″, run this command:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/usr/sbin/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime"
```

3. To rename the response ″E-mail root anytime″ to ″E-mail root and admin anytime″, run this command:

```
chresponse  -c "E-mail root and admin anytime" "E-mail root anytime"
```

These examples apply to management domains:

1. To delete the action named ″E-mail root″ from the response named ″E-mail root anytime″ that is defined on the management server, run this command on the management server:

```
chresponse -p -n "E-mail root" "E-mail root anytime"
```

2. In this example, the action named ″E-mail root″ will be used Monday through Friday from 8 AM to 6 PM, will use the command **/usr/sbin/rsct/bin/notifyevent root**, will save standard output in the audit log, and will expect return code 5 from the action. To add ″E-mail root″ to the response ″E-mail root anytime″ that is defined on the management server, run this command on the management server:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/usr/sbin/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime"
```

3. To delete the action named ″E-mail root″ from the response named ″E-mail root anytime″ that is defined on the managed node **nodeB**, run this command on the management server:

```
chresponse -p -n "E-mail root" "E-mail root anytime":nodeB
```

These examples apply to peer domains:

1. In this example, the action named ″E-mail root″ will be used Monday through Friday from 8 AM to 6 PM, will use the command **/usr/sbin/rsct/bin/notifyevent root**, will save standard output in the audit log, and will expect return code 5 from the action. To add ″E-mail root″ to the response ″E-mail root anytime″ that is defined on node **nodeA** in the domain, run this command on any node in the domain:

```
chresponse  -a  -n "E-mail root" -d 2-6 -t 0800-1800 \
-s "/usr/sbin/rsct/bin/notifyevent root" -o  -r 5  \
"E-mail root anytime":nodeA
```

2. To delete the action named ″E-mail root″ from the response named ″E-mail root anytime″ that is defined on node **nodeA** in the domain, run this command on any node in the domain:

```
chresponse -p -n "E-mail root" "E-mail root anytime":nodeA
```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **lscondresp**, **lsresponse**, **mkcondresp**, **mkresponse**, **rmresponse**

lscondition

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

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 of the information about all conditions is listed, the default format is long. If you specify one of the monitoring-status flags (**-e**, **-m**, or **-n**), the conditions with the specified 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. 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

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.

Flags

–a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lscondition -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–m Lists only those conditions that are being monitored without error.

–n Lists only those conditions that are not being monitored.

–e Lists only those conditions that are monitored in error.

–C Displays 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.

–l Produces long-formatted output. Displays the condition information on separate lines.

–t Displays the condition information in separate columns (table format).

–d Produces delimiter-formatted output. The default delimiter is a colon (**:**). Use the **–D** flag if you want to change the default delimiter.

–D *delimiter*
Produces delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–A Displays all of the attributes of the condition.

–q Does not return an error when the condition does not exist.

–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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lscondition

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"
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **chcondition**, **lscondresp**, **mkcondition**, **rmcondition**

lscondresp

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 pair, specify 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.

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 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.

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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lscondresp

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"
```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **mkcondition**, **mkcondresp**, **mkresponse**, **rmcondresp**, **startcondresp**, **stopcondresp**

lsevent

Purpose

lsevent Lists event monitoring information from the audit log.

Syntax

To list the 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 events or responses, or both, for a condition from the audit log:

lsevent [**–R** | **–A**] [**–O** *entries*] [**–B** *MMddhhmmyyyy*] [**–E** *MMddhhmmyyyy*] [**–e a** | **r** | **b**] [**–i**] [**–a** | **n** *node1*[**,***node2*...]] [**–w** *event_node*] [**–h**] [**–TV**] *condition* [response [response...]]

To list responses from the audit log:

lsevent -r [**–O** *entries*] [**–B** *MMddhhmmyyyy*] [**–E** *MMddhhmmyyyy*] [**–e a** | **r** | **b**] [**–i**] [**–a** | **n** *node1*[**,***node2*...]] [**–h**] [**–TV**] [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 the Event Response Resource Manager subsystem of RSCT. By using lsevent, you can list audit log information without knowing detailed information about the ERRM subsystem audit log templates as you would need using the **lsaudrec** command.

By default, without using options and operands, **lsevent** lists the events that are recorded in the audit log. These describe the monitored events that have 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 have 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 the events, the rearm events, or both 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 of time. 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 **lsevent** will retrieve 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 is the time that 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 will default 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
Specifies the type of event to list from the audit log:

a Lists events from conditions. This is the default.

r Lists rearm events from conditions.

b List both events and rearm events from conditions.

-E *MMddhhmmyyyy*
Specifies to list the audit log entries up to or ending at the time indicated. This is the time that 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 will default 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 on event registration, event errors, and responses about to be run.

n *node1***[,***node2***...]**
Specifies the node or nodes to 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 will be 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** will cause **lsevent** 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'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.

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's 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

The user of the **lsevent** command 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 Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsevent

Examples

1. To list the information for events that have 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
   ```

Related information

“rmccli ” on page 74, for general information about RMC-related commands

Books: *RSCT: Administration Guide*, for information about ERRM operations

Commands: **lsaudrec**, **lscondition**, **lscondresp**, **lsresponse**, **mkcondition**, **mkresponse**, **startcondresp**, **stopcondresp**

lsresponse

Purpose

Lists information about one or more responses.

Syntax

lsresponse [**–a**] [**–C** | **–l** | **–t** | **–d** | **–D** *delimiter*] [**–A**] [**–q**] [**–U**] [**–x**] [**–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.

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.*

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.

Flags

–a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lsresponse -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

–C Displays the **mkresponse** command that can be used to create the response and one of its actions. If more than one response is specified, each **mkresponse** command appears on a separate line. This flag is ignored when no responses are specified. This flag overrides the **–l** flag.

–l Displays the response information on separate lines (long form).

–t Displays the response information in separate columns (table form).

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **–D** flag if you wish to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

–A Displays all of the attributes of the response.

–q Does not return an error when **response** does not exist.

–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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsresponse

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     = "/usr/sbin/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     = "/usr/sbin/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"
```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **chresponse**, **lscondresp**, **mkcondresp**, **mkresponse**, **rmresponse**

mkcondition

Purpose

Creates a new condition definition which can be monitored.

Syntax

mkcondition **–r** *resource_class* **–e***″event_expression″* [**–E***″rearm_expression″*]
[**–d** *″event_description″*] [**–D** *″rearm_description″*]
[**–m l** | **m** | **p**] [**–n** *node_name1*[**,***node_name2*...]]
[**–p** *node_name*] [**––qnotoggle** | **––qtoggle**]
[**–s** *″selection_string″*] [**–S c** |**w** |**i**]
[**–h**] [**–TV**] *condition*

mkcondition **–c** *existing_condition*[**:***node_name*] [**–r** *resource_class*]
[**–e** *″event_expression″*] [**–E** *″rearm_expression″*]
[**–d** *″event_description″*] [**–D** *″rearm_description″*]
[**–m l** | **m** | **p**][**–n** *node_name1*[**,***node_name2*...]]
[**–p** *node_name*][**––qnotoggle** | **––qtoggle**]
[**–s** *″selection_string″*] [**–S c** | **w** | **i**]
[**–h**] [**–TV**] *condition*

Description

The **mkcondition** command creates a new condition with the name specified by the *condition* parameter. The condition is used to monitor a resource for the occurrence of the condition (or event). 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.

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 that it cannot be modified or removed, use the **chcondition** command with the **-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. This applies to the **-n** flag only. 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 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.

Flags

–c *existing_condition*[**:***node_name*]
: Copies an existing condition. The existing condition is defined on *node_name*. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable. If any other flags are specified, update the new condition as indicated by the flags. Links with responses are not copied.

–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 a dynamic attribute of *resource_class*, a mathematical comparison symbol (**>**, for example), and a constant.

–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 *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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mkcondition

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** man page for more information.

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 d \
"FileSystem space used"
```

3. To define a condition with the name ″nodeB FileSystem space used″ on **nodeB** to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the condition with local scope, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -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"
```

Related information

“rmccli ” on page 74

Books:

- *CSM: Administration Guide*, for information about node groups
- *CSM: Command and Technical Reference*, for information about the **nodegrp** command
- *RSCT: Administration Guide*, for information about ERRM operations and about how to use expressions and selection strings

Commands: **chcondition**, **lscondition**, **mkcondresp**, **mkresponse**, **nodegrp**, **rmcondition**, **startcondresp**

mkcondresp

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 with one of these commands: **rmcondresp**, **startcondresp**, or **stopcondresp**.

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*.

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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mkcondresp

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"
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations and about how to use expressions and selection strings

Commands: **lscondresp**, **mkcondition**, **mkresponse**, **rmcondresp**, **startcondresp**, **stopcondresp**

mkresponse

Purpose

Creates a new response definition.

Syntax

To create a response with no actions:

mkresponse [**–p** *node_name*] [**–h**] [**–TV**] *response*

To create a response with one action:

mkresponse –n *action* [**–d** *days_of_week*[**,***days_of_week...*]]
[**–t** *time_of_day*[**,***time_of_day...*]] **–s** *action_script* [**–r** *return_code*]
[**–e a** | **r** | **b**] [**–o**] [**–E** *env_var*=*value*[**,***env_var*=*value...*]]
[**–u**] [**–p** *node_name*] [**–h**] [**–TV**] *response*

To copy a response:

mkresponse –c *existing_response*[**:***node_name*] [**–p** *node_name*] [**–h**] [**–TV**]
response

Description

The **mkresponse** command creates a new response definition with the name specified by the *response* parameter. One action can also be specified when the response is defined. Actions define commands to be run when the response is used with a condition and the condition occurs. The action defines days of the week when the action can be used, the time of day for those days of the week, the script or command to be run, what type of event causes the command to be run, the expected return code of the script or command, and whether to keep standard output. The days and times are paired so that different times can be specified for different days. A response with no actions only logs the events.

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 that it cannot be modified or removed, use the **chresponse** command with the **-L** flag.

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.

Flags

–c *existing_response*[**:***node_name*]
Copies an existing response. Links with conditions are not copied. The existing response is defined on the node known as *node_name* in a 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.

–n *action*
Specifies the name of the action being defined. Only one action can be defined when the response is created. Use the **chresponse** command to add more actions to the response.

–d *days_of_week*

Specifies the days of the week when 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

–t *time_of_day*
Specifies the time range when *action* can be run, consisting of the start time followed by the end time, separated by a hyphen. *days_of_week* and *time_of_day* together define the interval when the action can be run.

The time is in 24–hour format (*hhmm*) where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. More than one *time_of_day* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default value is 0000-2400. If no value is specified but a comma is entered, the default value is used.

–s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the man pages for **logevent**, **notifyevent**, and **wallevent** for descriptions of the predefined response scripts provided with the application.

–r *return_code*
Specifies the expected return code for *action_script*. If the expected return code is specified, the actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating

whether they match. If the **–r** flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

–e a | r | b
Specifies the type of event that causes the action being defined to run:

a Event. This is the default.

r Rearm event.

b Both event and rearm event.

–o Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

–E *env_var*=*value*[,*env_var*=*value*...]
Specifies any environment variables to be set before running the action. If multiple *env_var*=*value* variables are specified, they must be separated by commas.

–u Specifies that the action is to be run when a monitored resource becomes undefined.

–p *node_name*
Specifies the name of the node where the response is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

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.

–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.

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'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.

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.

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 *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mkresponse

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 **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
"E-mail root 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 **/usr/sbin/rsct/bin/notifyevent root** for events, run this command:

```
mkresponse -n "E-mail root"     \
-d 1+7,2-6  -t 0000-2400,0800-1700  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e a  \
"E-mail root 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 **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, and that sets the environment variable LANG to en_US, run this command:

```
mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b \
-E LANG="en_US"  "E-mail root 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 **/usr/sbin/rsct/bin/notifyevent root** for rearm events, and that saves standard output in the audit log, expecting return code 5, run this command:

```
mkresponse -n "E-mail root" -d 2-6  -t 0800-1800  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e r  -o   \
-r 5  "E-mail root first shift"
```

6. To define a response with the name ″Critical notifications″ as a copy of ″Warning notifications″, enter:

```
mkresponse -c "Warning notifications" "Critical notifications"
```

These examples apply to management domains:

1. To define a response on the management server with the name ″E-mail root anytime″ that has an action named ″E-mail root″, to be used any time Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
"E-mail root 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 **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, run this command on the management server:

```
 mkresponse -n "E-mail root" -d 1+7  \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
-p nodeB "E-mail root 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 **/usr/sbin/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 "/usr/sbin/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 **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, and that sets two environment variables (PAGE ALL and TIMER SET), run this command from any node in the domain:

   ```
    mkresponse -n "E-mail root" -d 1+7  \
   -s "/usr/sbin/rsct/bin/notifyevent root" -e b   \
   -p nodeB -E 'ENV1="PAGE ALL", ENV2="TIMER SET"' \
   "E-mail root 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"
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **chresponse**, **lsresponse**, **mkcondition**, **mkcondresp**, **rmresponse**, **startcondresp**

rmcondition

Purpose

Removes a condition.

Syntax

rmcondition [**–f**] [**–q**] [**–h**] [**–TV**] *condition*[**:***node_name*]

Description

The **rmcondition** command removes the condition specified by the *condition* parameter. The condition must already exist to be removed. When the condition must be removed even if it has linked responses, use the **-f** flag to force the condition and the links with the responses to be removed. If the **-f** flag is not specified and links with responses exist, the condition is not removed. This command does not remove responses.

If a particular condition is needed for system software to work properly, it may be locked. A locked condition cannot be modified or removed until it is unlocked. If the condition you specify on the **rmcondition** command is locked, it will not be removed; instead an error will be generated informing you that the condition is locked. To unlock a condition, you can use the **-U** flag of the **chcondition** command. However, since a condition is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

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 to be removed.

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.

Flags

–f Forces the condition to be removed even if it is linked to responses. The links with the responses are removed as well as the condition, but the responses are not removed.

–q Does not return an error when *condition* does not exist.

–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.

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'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.

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.

Security

The user needs write permission for the **IBM.Condition** resource class to run **rmcondition**. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmcondition

Examples

These examples apply to standalone systems:

1. To remove the condition definition named ″FileSystem space used″, run this command:

   ```
   rmcondition "FileSystem space used"
   ```

2. To remove the condition definition named ″FileSystem space used″ even if the condition is linked with responses, run this command:

   ```
   rmcondition -f "FileSystem space used"
   ```

This example applies to management domains:

1. In this example, the current node is the management server. To remove the condition definition named ″nodeB FileSystem space used″ that is defined on managed node **nodeB**, run this command:

   ```
   rmcondition "FileSystem space used:nodeB"
   ```

This example applies to peer domains:

1. To remove the condition definition named ″nodeA FileSystem space used″ that is defined on node **nodeA**, run this command from any node in the domain:

   ```
   rmcondition "nodeA FileSystem space used:nodeA"
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **chcondition**, **lscondition**, **lscondresp**, **mkcondition**

rmcondresp

Purpose

Deletes the link between a condition and one or more responses.

Syntax

To delete the link between a condition and one or more responses:

rmcondresp [**–q**] [**–h**] [**–TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To delete all of the links to one or more responses:

rmcondresp [**–q**] **–r** [**–h**] [**–TV**] *response1* [*response2*...][**:***node_name*]

To lock or unlock the condition/response association:

rmcondresp { **-L** | **-U** } [**–h**] [**–TV**] *condition*[**:***node_name*] *response*

Description

The **rmcondresp** command deletes the link between a condition and one or more responses. A link between a condition and a response is called a *condition/response association.* The response is no longer run when the condition occurs. Use the **-r** flag to specify that the command parameters consist only of responses. This deletes all links to conditions for these responses. If only a condition is specified, links to all responses for that condition are deleted.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be removed by the **rmcondresp** command. If the condition/response association you specify on the **rmcondresp** command is locked, it will not be removed; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

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 the condition linked to the response. The condition is always specified first unless the **-r** flag is used.

response Specifies the name of a response or more than one response. The links from the specified responses to the specified condition are removed.

node_name Specifies the node where the condition is defined. If the **-r** flag is used, it is the node where the response is defined. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Flags

–q Does not return an error when either *condition* or *response* does not exist.

–r Indicates that all command parameters are responses. There are no conditions specified. This command removes condition/response associations from all conditions that are linked to the specified responses.

–L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

–U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

–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.

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'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.

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.

Security

The user needs write permission for the **IBM.Association** resource class to run **rmcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmcondresp

Examples

These examples apply to standalone systems:

1. To delete the link between the condition ″FileSystem space used″ and the response ″Broadcast event on-shift″, run this command:

   ```
   rmcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To delete the links between the condition ″FileSystem space used″ and all of its responses, run this command:

   ```
   rmcondresp "FileSystem space used"
   ```

3. To delete the links between the condition ″FileSystem space used″ and the responses ″Broadcast event on-shift″ and ″E-mail root anytime″, run this command:

   ```
   rmcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

4. To delete the links between the response ″Broadcast event on-shift″ and all of the conditions that use it, run this command:

   ```
   rmcondresp -r "Broadcast event on-shift"
   ```

These examples apply to management domains:

1. To delete the link between the condition ″FileSystem space used″ on the management server and the response ″Broadcast event on-shift″, run this command on the management server:

   ```
   rmcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To delete the links between the condition ″FileSystem space used″ on the managed node **nodeB** and the responses ″Broadcast event on-shift″ and ″E-mail root anytime″, run this command on the management server:

   ```
   rmcondresp  "FileSystem space used":nodeB \
   "Broadcast event on-shift" "E-mail root anytime"
   ```

These examples apply to peer domains:

1. To delete the links between the condition ″FileSystem space used″ on **nodeA** in the domain and the responses ″Broadcast event on-shift″ and ″E-mail root anytime″, run this command on any node in the domain:

   ```
   rmcondresp "FileSystem space used":nodeA \
   "Broadcast event on-shift" "E-mail root anytime"
   ```

2. To delete the links between all conditions on **nodeA** in the domain and the response ″Broadcast event on-shift″, run this command on any node in the domain:

   ```
   rmcondresp -r "Broadcast event on-shift":nodeA
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **lscondresp**, **mkcondition**, **mkcondresp**, **mkresponse**, **startcondresp**, **stopcondresp**

rmresponse

Purpose

Removes a response.

Syntax

rmresponse [**–f**] [**–q**] [**–h**] [**–TV**] *response*[**:***node_name*]

Description

The **rmresponse** command removes the response specified by the *response* parameter. The response must already exist in order to be removed. When the response must be removed even if it is linked with conditions, specify the **-f** flag. This forces the response and the links with the conditions to be removed. If the **-f** flag is not specified and links with conditions exist, the response is not removed. This command does not remove conditions.

If a particular response is needed for system software to work properly, it may be locked. A locked response cannot be modified or removed until it is unlocked. If the response you specify on the **rmresponse** command is locked, it will not be removed; instead an error will be generated informing you that the response is locked. To unlock a response, you can use the **-U** flag of the **chresponse** command. However, since a response is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it.

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

response Specifies the name of a defined response to be removed.

node_name Specifies the node in a cluster 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.

Flags

–f Forces the response to be removed even if it is linked with conditions. The links with the conditions are removed as well as the response, but the conditions are not removed.

–q Does not return an error when *response* does not exist.

–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.

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'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.

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.

Security

The user needs write permission for the **IBM.EventResponse** resource class to run **rmresponse**. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmresponse

Examples

These examples apply to standalone systems:

1. To remove the response definition named ″Broadcast event on-shift″, run this command:

   ```
   rmresponse  "Broadcast event on-shift"
   ```

2. To remove the response definition named ″Broadcast event on-shift″ even if the response is linked with conditions, run this command:

   ```
   rmresponse -f "Broadcast event on-shift"
   ```

This example applies to management domains:

1. In this example, the current node is the management server. To remove the response definition named ″Broadcast event on-shift″ on managed node **nodeB**, run this command:

   ```
   rmresponse  "Broadcast event on-shift":nodeB
   ```

This example applies to peer domains:

1. To remove the response definition named ″Broadcast event on-shift″ defined on node **nodeA**, run this command from any node in the domain:

   ```
   rmresponse  "Broadcast event on-shift":nodeA
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **chresponse**, **lscondresp**, **lsresponse**, **mkcondresp**, **mkresponse**

startcondresp

Purpose

Starts monitoring a condition that has one or more linked responses.

Syntax

To start monitoring a condition:

startcondresp [**–h**] [**–TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To lock or unlock the condition/response association:

startcondresp { **-L** | **-U** } [**–h**] [**–TV**] *condition*[**:***node_name*] *response*

Description

The **startcondresp** command starts the monitoring of a condition that has a linked response. A link between a condition and a response is called a *condition/response association.* In a cluster environment, the condition and the response must be defined on the same node. After monitoring is started, when the condition occurs, the response is run. If no responses are specified, monitoring is started for all responses linked to the condition. This causes all of the linked responses to run when the condition occurs. If more than one response is specified, monitoring is started only for those linked responses.

If one or more responses are specified and the responses are not linked with the condition, the **startcondresp** command links the specified responses to the condition, and monitoring is started. Use the **mkcondresp** command to link a response to a condition without starting monitoring.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be started by the **startcondresp** command. If the condition/response association you specify on the **startcondresp** command is locked, it will not be started; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition/response association so it cannot be started, stopped, or removed, reissue this command with the **-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.*

Parameters

condition Specifies the name of the condition linked to the response. The condition is always specified first.

node_name Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is

a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response Specifies the name of one or more responses. Specifying more than one response links the responses to the condition if they are not already linked and starts monitoring for the specified responses.

Flags

–L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

–U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

–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.

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'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.

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.

Security

The user needs write permission for the **IBM.Association** resource class to run **startcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/startcondresp

Examples

These examples apply to standalone systems:

1. To start monitoring for the condition ″FileSystem space used ″ by using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command:

   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

2. To start monitoring for the condition ″FileSystem space used ″ by using all of its linked responses, run this command:

   ```
   startcondresp "FileSystem space used"
   ```

3. To start monitoring for the condition ″FileSystem space used ″ by using the response ″Broadcast event on-shift″ and ″E-mail root anytime″, whether or not they are linked with the condition, run this command:

   ```
   startcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

These examples apply to management domains:

1. To start monitoring for the condition ″FileSystem space used″ on the management server using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command on the management server:

   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

2. To start monitoring for the condition ″FileSystem space used″ on the managed node **nodeB** using the response ″Broadcast event on-shift″, whether or not the response is linked with the condition, run this command on the management server:

```
startcondresp  "FileSystem space used":nodeB "Broadcast event on-shift"
```

This example applies to peer domains:

1. To start monitoring for the condition ″FileSystem space used″ on **nodeA** in the domain using the response ″Broadcast event on-shift″ (also on **nodeA** in the domain), whether or not the response is linked with the condition, run this command on any node in the domain:

```
startcondresp  "FileSystem space used":nodeA "Broadcast event on-shift"
```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **lscondresp**, **mkcondition**, **mkcondresp**, **mkresponse**, **stopcondresp**

stopcondresp

Purpose

Stops the monitoring of a condition that has one or more linked responses.

Syntax

To stop monitoring a condition:

stopcondresp [**–q**] [**–h**] [**–TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To lock or unlock the condition/response association:

stopcondresp { **-L** | **-U** } [**–h**] [**–TV**] *condition*[**:***node_name*] *response*

Description

The **stopcondresp** command stops the monitoring of a condition that has one or more linked responses. If no response is specified, all of the linked responses for the condition are stopped. If one or more responses is specified, only those responses that are linked to the condition are stopped. When the condition occurs, the response is not run. If no responses are active for a condition, the condition is no longer monitored.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be stopped by the **stopcondresp** command. If the condition/response link you specify on the **stopcondresp** command is locked, it will not be stopped; instead an error will be generated informing you that the condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. A condition/response association is typically locked because it is essential for system software to work properly, so you should exercise caution before unlocking it.

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 the condition linked to the response. The condition is always specified first.

node_name Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response Specifies the names of one or more responses. Monitoring is stopped for the specified responses. (If a specified response is not linked to the condition, it is ignored.)

Flags

–q Does not return an error when either *condition* or *response* does not exist or when the *condition* linked with *response* is not being monitored.

–U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

–L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

–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.

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'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.

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.

Security

The user needs write permission for the **IBM.Association** resource class to run **stopcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/stopcondresp

Examples

These examples apply to standalone systems:

1. To stop monitoring for the condition ″FileSystem space used ″ which has the response ″Broadcast event on-shift″ linked with it, run this command:

   ```
   stopcondresp "FileSystem space used" "Broadcast event on-shift"
   ```

2. To stop monitoring for the condition ″FileSystem space used ″ using all of its linked responses, run this command:

   ```
   stopcondresp  "FileSystem space used"
   ```

This example applies to management domains:

1. To stop monitoring for the condition ″FileSystem space used ″ on the managed node **nodeB** which has the response ″Broadcast event on-shift″ linked with it, run this command on the management server:

   ```
   stopcondresp "FileSystem space used:nodeB" "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To stop monitoring for the condition ″FileSystem space used ″ on the node **nodeA** which has the response ″Broadcast event on-shift″ linked with it, run this command on any node in the domain:

   ```
   stopcondresp "FileSystem space used:nodeA" "Broadcast event on-shift"
   ```

Related information

“rmccli ” on page 74

Books:

- *RSCT: Administration Guide*, for information about ERRM operations

Commands: **lscondresp**, **mkcondition**, **mkcondresp**, **mkresponse**, **startcondresp**

Chapter 9. ERRM scripts

© Copyright IBM Corp. 2002, 2007

elogevent, logevent

Purpose

Logs event information that the event-response resource manager (ERRM) generates to a specified log file.

Syntax

elogevent [**–h**] *log_file*

logevent [**–h**] *log_file*

Description

The **elogevent** script always return messages in English. The language in which the messages of the **logevent** script are returned depends on the locale settings.

When an event occurs, these scripts capture information about the event, which is posted by the event-response resource manager (ERRM), in environment variables that it (the ERRM) generates.

These scripts can be used as actions that an event-response resource runs. You can also use these scripts as templates to create other user-defined actions. To find out how an event-response resource runs an action command, see *RSCT: Administration Guide* .

These scripts return event information about the ERRM environment variables that are described in *RSCT: Administration Guide* . In addition, these scripts return the local time when the event is reported. **ERRM_TIME** is the environment variable that the ERRM supplies. The value of **ERRM_TIME** is localized and converted to readable form before it is displayed.

These scripts use the **alog** command to write event information to and read event information from the specified *log_file*.

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.

Flags

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

Standard output

When the **–h** flag is specified, the script's usage statement is written to standard output.

Exit status

0 The script has run successfully.

1 A required *log_file* is not specified.

2 The *log_file* path is not valid.

Restrictions

- These scripts must be run on the node where the ERRM is running.
- The user who runs these scripts must have write permission for the *log_file* where the event information is logged.

Implementation specifics

These scripts are part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/elogevent

/usr/sbin/rsct/bin/logevent

Examples

1. To log information, specify **/tmp/event.log**in the Web-based System Manager interface. ERRM runs this command:

   ```
   /usr/sbin/rsct/bin/logevent/tmp/event.log
   ```

   The **/tmp/event.log** file does not need to exist when the command is run.
2. To see the contents of the **/tmp/event.log** file, run this command:

   ```
   alog -f /tmp/event.log -o
   ```

   The following sample output shows a warning event for the **/var** file system (a file system resource):

   ```
   ============================================================================
   Event reported at Fri Feb 27 12:21:25 2004

   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
   ```

Related information

Commands: **alog**

Books: *RSCT: Administration Guide* , for information about the event-response resource manager (ERRM) and the ERRM environment variables

enotifyevent, notifyevent

Purpose

Sends information that the event-response resource manager (ERRM) generates about an event to a specified user ID.

Syntax

enotifyevent [**–h**] [*user-ID*]

notifyevent [**–h**] [*user-ID*]

Description

The **enotifyevent** script always return messages in English. The language in which the messages of the **notifyevent** script are returned depends on the locale settings.

When an event occurs, these scripts capture information about the event, which is posted by the event-response resource manager (ERRM), in environment variables that it (the ERRM) generates.

These scripts can be used as actions that an event-response resource runs. You can also use these scripts as templates to create other user-defined actions. To find out how an event-response resource runs an action command, see *RSCT: Administration Guide* .

These scripts return event information about the ERRM environment variables that are described in *RSCT: Administration Guide* . In addition, these scripts return the local time when the event is reported. **ERRM_TIME** is the environment variable that the ERRM supplies. The value of **ERRM_TIME** is localized and converted to readable form before it is displayed.

These scripts use the **mail** command to send event information to the specified user ID. When a user ID is specified, it is assumed to be valid, and it is used without verifying it. If a user ID is not specified, the user who is running the command is used as the default.

user-ID is the optional ID of the user to whom the event information will be mailed. If *user-ID* is not specified, the user who is running the command is used as the default.

Flags

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

Standard output

When the **–h** flag is specified, the script's usage statement is written to standard output.

Exit status

0 Command has run successfully.

Restrictions

1. These scripts must be run on the node where the ERRM is running.
2. The **mail** command is used to read the file.

Implementation specifics

These scripts are part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/enotifyevent

/usr/sbin/rsct/bin/notifyevent

Examples

1. Specify **user1** in Web-based System Manager to send mail to a user. The event-response resource manager then runs the following command:

   ```
   /usr/sbin/rsct/bin/notifyevent user1
   ```

2. You can use the **mail** command to read the contents of the event information. The following example shows how a warning event for the **/var** file system (a file system resource) is formatted and logged:

   ```
   ============================================================================
   Event reported at Fri Feb 27 12:21:25 2004

   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
   ```

Related information

Commands: **mail**

Books: *RSCT: Administration Guide* , for information about the event-response resource manager (ERRM) and the ERRM environment variables

ewallevent, wallevent

Purpose

Broadcasts an event or a rearm event to all users who are logged in.

Syntax

ewallevent [**–c**] [**–h**]

wallevent [**–c**] [**–h**]

Description

The **ewallevent** script always return messages in English. The language in which the messages of the **wallevent** script are returned depends on the locale settings.

These scripts broadcast a message about an event or a rearm event to all users who are currently logged in to the host when the event or the rearm event occurs. When an event or a rearm event occurs, the event-response resource manager (ERRM) captures and posts event or rearm event information in environment variables that it (the ERRM) generates. These scripts can be used as actions that an event-response resource runs. You can also use these scripts as templates to create other user-defined actions.

To find out how an event-response resource runs an action command, see *RSCT: Administration Guide* .

Messages are displayed in the following format at the consoles of all users who are logged in when an event or a rearm event occurs for which the script is a response action:

```
Broadcast message from user@host (tty) at hh:mm:ss...

severity event_type occurred for Condition condition_name
on the resource resource_name of resource_class_name at hh:mm:ss mm/dd/yy
The resource was monitored on node_name and resided on {node_names}.
```

These scripts return event information about the ERRM environment variables that are described in *RSCT: Administration Guide* . In addition, these scripts return the local time when the event or the rearm event occurs. **ERRM_TIME** is the environment variable that the ERRM supplies. The value of **ERRM_TIME** is localized and converted to readable form before it is displayed.

These scripts capture the values of the ERRM environment variables and use the **wall** command to write a message to the user consoles that are currently logged in.

Flags

–c Instructs **wallevent** to broadcast the value of an ERRM event. When the **–c** flag is specified, **wallevent** broadcasts the messages that are contained in the **ERRM_VALUE** environment variable.

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

Standard output

When the **–h** flag is specified, the script's usage statement is written to standard output.

Exit status

0 Script has run successfully.

1 Error occurred when the script was run.

Restrictions

1. These scripts must be run on the node where the ERRM is running.
2. The **wall** command is used to write a message to currently logged-in user consoles. Refer to the **wall** man page for more information on the **wall** command.

Implementation specifics

These scripts are part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/ewallevent

/usr/sbin/rsct/bin/wallevent

Examples

1. Suppose the **wallevent** script is a predefined action in the critical-notification response, which is associated with the **/var space used** condition on the resource **/var**. The threshold of the event expression defined for this condition is met, and an event occurs. The critical-notification response takes place, and **wallevent** is run. The following message is displayed on the consoles of all users who are logged in:

   ```
   Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

   Critical event occurred for Condition /var space used
   on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
   The resource was monitored on c174n05 and resided on {c174n05}.
   ```

2. When a rearm event occurs for the **/var space used** condition on the resource **/var**, the following message is displayed on the consoles of all users who are logged in:

   ```
   Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

   Critical rearm event occurred for Condition /var space used
   on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
   The resource was monitored on c174n05 and resided on {c174n05}.
   ```

Related information

Commands: **wall**

Books: see *RSCT: Administration Guide* for information about the event-response resource manager (ERRM) and about how to use ERRM environment variables

snmpevent

Purpose

Sends ERRM events to an SNMP agent.

Syntax

snmpevent [**-a** *host-name*] [**–c** *community*] [**-h**]

Description

The **snmpevent** script sends a Simple Network Management Protocol (SNMP) trap of an event-response resource manager (ERRM) event to a host running an SNMP agent. The agent formats the trap information into an SNMP trap and sends it to the SNMP manager that is defined in its configuration file. This script is meant to be called by the predefined ERRM response **Generate SNMP trap**. When an event or a rearm event occurs, the event-response resource manager (ERRM) captures and posts information about the event or the rearm event in environment variables that it (the ERRM) generates.

You can also use the **snmpevent** script as a template to create other user-defined actions. To find out how an event-response resource runs an action command, see *RSCT: Administration Guide* .

A template of the message that is sent as a trap when an event or a rearm event occurs and **snmpevent** is the defined response follows:

```
ERRM_COND_SEVERITY ERRM_TYPE occurred:
  Condition:  ERRM_COND_NAME
  Node:  ERRM_NODE_NAME
  Resource:  ERRM_RSRC_NAME
  Resource Class:  ERRM_RSRC_CLASS_NAME
  Resource Attribute:  ERRM_ATTR_NAME
  Attribute Type:  ERRM_DATA_TYPE
  Attribute Value:  ERRM_VALUE
```

The **snmpevent** command captures the values of these environment variables and formats a generic message that is sent as a trap through a call to the **snmptrap** command.

Flags

–a *host-name*
: Specifies the host name of the SNMP agent to which the AIX subagent will connect. By default, the subagent will connect to the SNMP agent running on the local node.

–c
: Specifies the SNMP community to be used. This can be any string the SNMP agent will accept. The default is **public**.

–h
: Writes this script's usage statement to standard output.

Standard output

When the **–h** flag is specified, this script's usage statement is written to standard output.

Exit status

0
: The script has run successfully.

1 An error occurred when the script was run.

Restrictions

This script must be run on the node where the ERRM is running.

Implementation specifics

This script is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/snmpevent

Examples

1. Suppose the command **/usr/sbin/rsct/bin/snmpevent** is an action in the critical-notification response, which is associated with the CSM predefined condition **NodeChanged**. This can be done with the **mkcondresp** command followed by the **startcondresp** command. The **/etc/snmpdv3.conf** file should be configured to where the trap will be sent. In this example, if you want the trap sent to **9.117.16.246**, write the **/etc/snmpdv3.conf** file as follows:

```
VACM_GROUP group1 SNMPv1  public  -

VACM_VIEW defaultView        internet                    - included
-VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -

NOTIFY notify1 traptag trap -

#TARGET_ADDRESS Target1 UDP 127.0.0.1       traptag trapparms1 - - -
TARGET_ADDRESS Target1 UDP 9.117.16.246       traptag trapparms1 - - -

TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -

COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -

DEFAULT_SECURITY no-access - -

logging         file=/usr/tmp/snmpdv3.log       enabled
logging         size=0                          level=0

smux            1.3.6.1.4.1.2.3.1.2.1.2         gated_password  # gated

snmpd smuxtimeout=200 #muxatmd
smux 1.3.6.1.4.1.2.3.1.2.3.1.1 muxatmd_password #muxatmd
```

   Then, restart the **snmpd** daemon by first terminating the **snmpd** daemon that is currently running and then starting it again:

```
# ps -ef | grep snmpd
    root  4570 12956   1 08:24:32  pts/0  0:00 grep snmpd
    root 13810     1   0 08:11:04      -  0:00 snmpd
# kill -9 13810
# snmpd
```

   Next, change the **LParID** property of node **c175n08** to **12**:

```
# chnode c175n08 LParID=12
```

   Now, on the node **9.117.16.158** (the node with the SNMP manager that was specified in the **/etc/snmpdv3.conf** file), the SNMP manager should record something like this:

```
2004-02-27 12:21:25 c174tr1.ppd.pok.ibm.com [9.114.78.17] TRAP, SNMP v1,
community public
        enterprises.ibm Enterprise Specific Trap (1) Uptime: 0:01:45.00
        enterprises.ibm.ibmProd.191.1.6.1.0 = "Informational Event
occurred. Condition=NodeChanged Node=c174tr1.ppd.pok.ibm.com
Resource=c175n08.ppd.pok.ibm.com Resource Class=Node Resource
Attribute=Changed Attributes Attribute Type=CT_CHAR_PTR_ARRAY Attribute
Val={LParID} "
```

The output varies based on SNMP managers.

Related information

Commands: **lscondition**, **lscondresp**, **lsresponse**, **mkcondition**, **mkcondresp**, **mkresponse**, **snmptrap**, **startcondresp**, **stopcondresp**

Books: *RSCT: Administration Guide* , for information about the event-response resource manager (ERRM) and the ERRM environment variables

Chapter 10. Sensor resource manager commands

© Copyright IBM Corp. 2002, 2007

chsensor

Purpose

Changes the attributes of a resource monitoring and control (RMC) sensor.

Syntax

chsensor [**–a** | **–n** *host1* [**,** *host2* **,** ...] | **–N** { *node_file* | *"–"* }] [**-i** *seconds*] [**–h**] [**–v** | **–V**] *sensor_name attr1*=*value1* [*attr2*=*value2* ...]

Description

The **chsensor** command changes the attributes of a resource monitoring and control (RMC) sensor. Use the *sensor_name* parameter to specify which sensor you are changing.

The **chsensor** command runs on any node. If you want **chsensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **chsensor** 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*.

Parameters

sensor_name
Specifies the name of the sensor to change.

attr1=*value1* **[***attr2*=*value2* **...]**
Specifies one or more sensor attributes and the new values to which they will be set. You can change the values of these attributes:

Name Specifies the new name of the sensor. If the new name is a string that contains spaces or special characters, it must be enclosed in quotation marks.

ControlFlags Specifies that special handling is required for this sensor instead of the default behavior. You can specify one of these values:

0 Indicates that no special handling is required. This is the default. The command in this sensor will not run when **lssensor** is called or when monitoring is started.

1 Indicates that the command in this sensor will be run any time, even at the initial stage (when **lssensor** is called or when monitoring is just started.) It is not recommended that you specify this value, unless you expect the command to run very soon. Setting this value could block other requests to the sensor resource manager, so that those requests will not be processed until the command ends.

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 command when monitoring is 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**.

UserName Specifies the name of a user whose privileges will be used to run the command. The user should already be defined on the system.

Description Provides a description of the sensor and what it is monitoring.

ErrorExitValue

Specifies which exit values will be interpreted as errors, as follows:

0 No exit values are interpreted as errors.

1 Exit values other than **0** are interpreted as errors.

2 An exit value of **0** is interpreted as an error.

If the exit value indicates an error as specified by this attribute, no dynamic attribute values (except **ExitValue**) are updated.

Flags

–a Changes 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, **chsensor -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.

–i *seconds*

Specifies the interval in which the sensor command is run to update the values of the sensor attributes. *seconds* is an integer value and must be greater than or equal to **10**. The sensor command is run at the specified interval only when a sensor resource is monitored. If the interval is set to **0**, the sensor command will not be automatically run. Using the **refsensor** command is independent of interval updates.

–n *host1* **[,** *host2* **, ...]**

Specifies the node on which the sensor should be changed. By default, the

sensor is changed 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.

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

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

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.

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.

Security

The user needs write permission for the **IBM.Sensor** resource class in order to run **chsensor**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chsensor

Examples

1. To change the **Name** attribute of the **SensorA** sensor to **Sensor1A**, enter:

   ```
   chsensor SensorA Name=Sensor1A
   ```

2. To change the update interval of the **SensorA** sensor to **10**, enter:

   ```
   chsensor -i 10 SensorA
   ```

3. To change the **Name** attribute of the **SensorA** sensor to **Sensor1A** on the nodes listed in the **/u/joe/common_nodes** file, enter:

   ```
   chsensor -N /u/joe/common_nodes SensorA Name=Sensor1A
   ```

   where **/u/joe/common_nodes** contains:

   ```
   # common node file
   #
   node1.myhost.com    main node
   node2.myhost.com    backup node
   ```

Related information

“rmccli ” on page 74, for information about *attr*=*value* syntax

Books: *RSCT: Administration Guide*, for information about the ACL authorization file

Commands: **lssensor**, **mksensor**, **refsensor**, **rmsensor**

lssensor

Purpose

Displays the sensors that are defined to the resource monitoring and control (RMC) subsystem.

Syntax

lssensor [**–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 don't specify any sensor names, **lssensor** 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.

Using **lssensor -A** or **lssensor** *sensor_name* causes the sensor command to run. A *sensor command* is a command or script that is defined (using the **mksensor** command) to set the sensor attribute values. The **lssensor** command displays values for attributes that you can set using a sensor command, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values.

The **lssensor** command displays the following information about defined sensors:

Field	***Description***
Name	The name of the sensor
ActivePeerDomain	Indicates the peer domain for which information is being displayed
Command	The sensor command that is run to set or 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 sensor resource manager interprets the exit value
ErrorMessage	This field is not used **(??)**
ExitValue	The exit code from the sensor command that is running
Float32	The type **float32** attribute for this sensor
Float64	The type **float64** attribute for this sensor
Int32	The type **int32** attribute for this sensor
Int64	The type **int64** attribute for this sensor
NodeNameList	The name of the node where this sensor 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
TimeCommandRun	Indicates the date and time that the sensor command was run
Uint32	The type **uint32** attribute for this sensor
Uint64	The type **uint64** attribute for this sensor
UserName	The user ID that is used when run the sensor command is run

Use the **-l**, **-t**, **-d**, or **-D** flags to have the output displayed in long format, table format, or delimiter format. The **-x** flag will omit headings when any of these options 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*.

Parameters

sensor_name1 **[***sensor_name2***...]**
Specifies the names of one or more sensors to display.

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 long formatted output. Each attribute is displayed on a separate line.

–n *host1*[**,***host2*...]
Specifies the nodes 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.

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.

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.

Security

The user needs read permission for the **IBM.Sensor** resource class in order to run **lssensor**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lssensor

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
   ErrorMessage =
   ExitValue = 0
   Float32 =
   ```

```
Float64 =
Int32 =
Int64 =
NodeNameList = {node1.gumby.com}
RefreshInterval = 60
SavedData = Last SavedData
SD = [string from sensor1,1063814,1063814,1063814,1063814,1.06381e+06,1.06381e+06]
String =
TimeCommandRun = Tue Nov  6 10:27:19 2007
Uint32 =
Uint64 =
UserName = root
-------------------
Name = CFMRootModTime
ActivePeerDomain =
Command = /opt/csm/csmbin/mtime/cfmroot
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ErrorMessage =
ExitValue = 0
Float32 =
Float64 =
Int32 =
Int64 =
NodeNameList = {node2.gumby.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
String =
TimeCommandRun = Tue Nov  6 10:27:19 2007
Uint32 =
Uint64 =
UserName = root
-------------------
Name = ErrorLogSensor
ActivePeerDomain =
Command = /opt/csm/csmbin/monerrorlog
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ExitValue = 0
Float32 =
Float64 =
Int32 =
Int64 =
NodeNameList = {node3.gumby.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
String =
TimeCommandRun = Tue Nov  6 10:29:17 2007
Uint32 =
Uint64 =
UserName = root
-------------------
.
.
.
```

3. To list the attributes of **sensor2**, enter:

```
lssensor sensor2
```

The output will look like this:

```
Name = sensor2
ActivePeerDomain =
Command = /usr/local/bin/sensorcmd2
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ErrorMessage =
ExitValue = 127
Float32 =
Float64 =
Int32 =
Int64 =
NodeNameList = {node2.gumby.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
String =
TimeCommandRun = Tue Nov  6 10:31:39 2007
Uint32 =
Uint64 =
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:NodeNameList:RefreshInterval:SD:SavedData:String:TimeCommandRun:Uint32:Uint64:UserName:
JoeExample:JoeDomain:cat /etc/motd:0:0::1::0:::::{node1.myhost.com}:60:[,0,0,0,0,0,0]:::Fri Feb  2 19:00:00 2007:::
root:
JoeSample:JoeDomain:/usr/sbin/rsct/install/bin/ctversion:0:0::1::0:::::{node1.myhost.com}:60:[,0,0,0,0,0,0]:::
Fri Feb  2 19:00:00 2007:::root:
JoeSens:JoeDomain:/tmp/sensor/numusers:0:1::1::0:::::{node1.myhost.com}:0:[,2,0,0,0,0,0]:::Tue Nov  6 10:27:19 2007:
::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
```

Related information

Books: *RSCT: Administration Guide*, for information about the ACL authorization file

Commands: **chsensor**, **mksensor**, **refsensor**, **rmsensor**

mksensor

Purpose

Defines a sensor to the resource monitoring and control (RMC) subsystem.

Syntax

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*[*"*]

Description

The **mksensor** command defines a sensor to the RMC subsystem. A *sensor* is an RMC resource that has attributes you can monitor. You can use event-response resource manager (ERRM) commands to set up monitoring of the sensor attributes. The response actions you define will run when a monitored sensor event occurs. Using sensors, you can extend RMC's monitoring capabilities — without having to create a new resource manager.

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.

Alternatively, you can use **chsensor** or **refsensor** to update the sensor attribute values. The **lssensor** command displays values for sensor attributes that you can set using a sensor command, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. To remove a sensor, use the **rmsensor** command.

The sensor consists of the following attributes, which can be set by the sensor command:

Float32 The type float32 attribute for this sensor

Float64 The type float64 attribute for this sensor

Int32 The type int32 attribute for this sensor

Int64 The type int64 attribute for this sensor

Quantum The type quantum attribute for this sensor

String The type string attribute for this sensor

Uint32 The type uint32 attribute for this sensor

Uint64 The type uint64 attribute for this sensor

The 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.

The **mksensor** command runs on any node. To define a sensor 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*.

Parameters

sensor_name
Specifies the name of the sensor 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 interepreted 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.

Flags

–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 in which *sensor_command* is run to update the values of the sensor attributes. *seconds* is an integer value and must be greater than or equal to **10**. *sensor_command* is run at the specified interval only when a sensor resource is monitored. The default interval is **60** seconds. If the interval is set to **0**, *sensor_command* will not be run automatically. Using the **refsensor** command is independent of interval updates.

–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.

1 Indicates that the command in this sensor will be run any time, even at the initial stage (when **lssensor** is called or when monitoring is just started.) It is not recommanded that you specify this value, unless you expect the command to run very soon. Setting this value could block other requests to the sensor resource manager, so that those requests will not be processed until the command ends.

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 command when monitoring is 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.

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.

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.

Security

The user needs write permission for the **IBM.Sensor** resource class in order to run **mksensor**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

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 Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mksensor

Examples

1. To create a 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** could be something like this:

   ```
   #!/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 could 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
   ```

Related information

“rmccli ” on page 74, for information about *attr*=*value* syntax

Books: *RSCT: Administration Guide*, for information about the ACL authorization file and about monitoring resources with the ERRM commands

Commands: **chsensor**, **lssensor**, **mkcondition**, **mkresponse**, **refsensor**, **rmsensor**, **startcondresp**

refsensor

Purpose

Refreshes a sensor defined to the resource monitoring and control (RMC) subsystem.

Syntax

refsensor [**–a** | **–n** *host1*[**,***host2*...] | **–N** { *node_file* | *"–"* }] [**–h**] [**–v** | **–V**] *sensor_name* [*attr1***=***value1* [*attr2***=***value2*] ...]

Description

The **refsensor** command refreshes a sensor resource that is defined to the resource monitoring and control (RMC) subsystem. A sensor is an RMC resource that has attributes that can be monitored. The sensor can be refreshed using **refsensor** in one of two ways: either by running the sensor command that is defined for the sensor resource or by specifying values for specific sensor attributes. The sensor must be monitored for **refsensor** to run successfully.

The **refsensor** command runs on any node. If you want **refsensor** to run on all of the nodes in a domain, use the **-a** flag. If you want refsensor 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*.

To have **refsensor** update specific sensor attributes, specify one or more *attr***=***value* parameters. Only the attributes specified will be updated. No other sensor attributes will be updated. The sensor attributes that can be specified as parameters are:

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.

For example, to update the sensor attributes **Int32** and **Float32** only for the sensor named **Sensor1**, enter:

```
refsensor Sensor1 Int32=45 Float32=7.8
```

No other sensor attributes will be updated.

When the **refsensor** command runs, it does not affect the interval, if any, that is defined for running the sensor command. That is, if a monitored sensor is being updated every 60 seconds, running **refsensor** does not cause the interval timer to be reset back to 60 seconds.

Parameters

sensor_name
Specifies the name of the sensor to be refreshed.

attr=*value*
Specifies which sensor attributes will be refreshed and the values to which they will be set.

Flags

–a Refreshes 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, **refsensor -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.

–n *host1*[**,***host2*...]
Specifies one or more nodes on which the sensor should be refreshed. By default, the sensor is refreshed 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.

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

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

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.

Exit status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

4 The sensor is not monitored and cannot be refreshed.

6 No sensor resources were found.

n Based on other errors that can be returned by the RMC subsystem.

Security

The user needs write permission for the **IBM.Sensor** resource class in order to run **refsensor**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/refsensor

Examples

1. To refresh the sensor called **Sensor1** so that its defined sensor command is run, run this command:

   ```
   refsensor Sensor1
   ```

2. To refresh the sensor called **Sensor1** so that **Int32** is set to **50**, **Float32** is set to **123.45**, and **String** is set to ″**test input**″, run this command:

   ```
   refsensor Sensor1 Int32=50 Float32=123.45 String="test input"
   ```

3. To refresh the sensor called **Sensor1** on the nodes that are listed in the **/u/joe/common_nodes** file so that **Sensor1**'s defined sensor command is run, run this command:

```
refsensor -N /u/joe/common_nodes Sensor1
```

where **/u/joe/common_nodes** contains:

```
# common node file
#
node1.myhost.com    main node
node2.myhost.com    backup node
```

Related information

"rmccli " on page 74, for information about *attr*=*value* syntax

Books: *RSCT: Administration Guide*, for information about the ACL authorization file

Commands: **chsensor**, **lssensor**, **mksensor**, **rmsensor**

rmsensor

Purpose

Removes a sensor from the resource monitoring and control (RMC) subsystem.

Syntax

rmsensor [**–a** | **–n** *host1*[**,***host2*...] | **–N** { *node_file* | *"–"* }] [**–h**] [**–v** | **–V**] *sensor_name1* [*sensor_name2*...]

Description

The **rmsensor** command removes one or more sensors from the sensor resource class in the resource monitoring and control (RMC) subsystem.

If the sensor is being monitored, monitoring will be stopped, but the ERRM resources defined for monitoring are not removed. To remove the ERRM resources, use the **rmcondition**, **rmresponse**, or **rmcondresp** command against the monitoring resources that were used for this sensor.

The **rmsensor** command runs on any node. If you want **rmsensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **rmsensor** 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*.

Parameters

sensor_name1 **[***sensor_name2***...]**
Specifies the names of one or more sensors to remove.

Flags

–a Removes 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, **rmsensor -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.

–n *host1***[,***host2***...]**
Specifies one or more nodes from which the sensor should be removed. By default, the sensor is removed 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.

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

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

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 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.

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.

Security

The user needs write permission for the **IBM.Sensor** resource class in order to run **rmsensor**. 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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmsensor

Examples

1. To remove the sensor called **sensor1**, enter:

   ```
   rmsensor sensor1
   ```

2. To remove the sensor called **sensor1** from the nodes that are listed in the **/u/joe/common_nodes** file, enter:

   ```
   rmsensor -N /u/joe/common_nodes sensor1
   ```

   where **/u/joe/common_nodes** contains:

   ```
   # common node file
   #
   node1.myhost.com    main node
   node2.myhost.com    backup node
   ```

Related information

Commands: **chsensor**, **lssensor**, **mksensor**, **refsensor**, **rmcondition**, **rmresponse**, **rmcondresp**

Books: *RSCT: Administration Guide*, for information about the ACL authorization file

Part 5. Auditing resources

Chapter 11. Audit log resource manager commands 291
lsaudrec 292
rmaudrec 298

© Copyright IBM Corp. 2002, 2007

Chapter 11. Audit log resource manager commands

© Copyright IBM Corp. 2002, 2007

lsaudrec

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

You can use the **lsaudrec** command 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 *RSCT: Administration Guide*.

You can also specify field names as parameters to this command to choose which fields are displayed and the order in which they are displayed. The common field names are:

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 that are described in the *RSCT: Administration Guide*, you can use the following syntax for dates and times with this command:

#*mmddhhmmyyyy*
: This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour, *mm* = minutes, *yyyy* = year. For example, **#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.

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.

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 *RSCT: Administration Guide*.

The names of fields in the record can be used in the expression. If the **-S** flag is not specified, only the names of common fields can be used. See the **Description** for a list of the common field names and their data types. If the **-S** flag is specified, the name of any field for the specified subsystem as well as the common field names can be used.

If this flag is omitted, the records that are displayed will depend on the **-S** flag. If the **-S** flag is omitted, all records from the audit log are displayed. Otherwise, all records for the subsystem identified by the **-S** flag are displayed.

-x Excludes the header (suppresses header printing).

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

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.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

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.

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.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lsaudrec

Examples

1. To list all records in the audit log on every node in the current management scope as determined by the CT_MANAGMENT_SCOPE environment variable, enter:

   ```
   lsaudrec
   ```

2. To list all records that were logged in the last hour on every node in the current management scope as determined by the CT_MANAGMENT_SCOPE environment variable, enter:

   ```
   lsaudrec -s "Time > #-000001"
   ```

3. To list the time and sequence number of every record in the audit log for the subsystem **abc** on nodes **mynode** and **yournode**, enter:

   ```
   lsaudrec -n mynode,yournode -S abc Time SequenceNumber
   ```

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 -s"ConditionName=='Condition1' && Etype==92"
   ```

8. To list error records only, enter:

   ```
   lsaudrec -s"Category=1"
   ```

9. 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.
```

10. 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
```

Related information

Books: *RSCT: Administration Guide*, for information about:

- how to specify selection strings
- using constants in expressions

Commands: **lsevent**, **rmaudrec**

rmaudrec

Purpose

Removes records from the audit log.

Syntax

rmaudrec [**–a** | **–n** *node_name1*[**,***node_name2*]...] [**–S** *subsystem_name*] **–s** *selection_string* [**–h**] [**–V**]

Description

You can use the **rmaudrec** command to delete 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. In addition, records may sometimes need to be removed explicitly, which can be done 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 deleted. The selection string is matched against each record using the referenced fields of each record to perform the match. Any records that match will be removed. The selection string is specified with the **-s** flag.

A selection string is an expression composed of field names, constants, and operators. The syntax of a selection string is very similar to an expression in the C programming language. For information on how to specify selection strings, see the *RSCT: Administration Guide*.

The common field names are:

Field	*Description*
Time	Specifies the time when the situation occurred that the record corresponds to. The value is a 64-bit integer and represents the number of microseconds since Unix Epoch (00:00:00 GMT January 1, 1970). See the constants below for specifying the time in more user-friendly formats.
Subsystem	Specifies the subsystem that generated the record. This is a string.
Category	Indicates the importance of the situation corresponding to the audit record, as determined by the subsystem that generated the record. The valid values are: **0** (informational) and **1** (error).
SequenceNumber	Specifies the unique 64-bit integer that is assigned to the record. No other record in the audit log will have the same sequence number.
TemplateId	Specifies the subsystem-dependent identifier that is assigned to records that have the same content and format string. This value is a 32-bit unsigned integer.
NodeName	Specifies the name of the node from which the record was obtained. This field name cannot be used in a selection string.

In addition to the constants in expressions that are described in the *RSCT: Administration Guide*, you can use the following syntax for dates and times with this command:

#*mmddhhmmyyyy*
: This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour, *mm* = minutes, *yyyy* = year. For example, **#010523042002** corresponds to January 5, 11:04 PM, 2002. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 0, hour = 0, day = 1, and month = the current month.

#-*mmddhhmmyyyy*
: This format is similar to the previous one, but is relative to the current time and date. For example, the value **#-0001** corresponds to one day ago and the value **#-010001** corresponds to one month and one hour ago. Fields can be omitted starting from the right and are replaced by 0.

The audit records considered for deletion 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 deletion can be restricted to a set of specific nodes by using the **-n** flag. If this flag is specified, the search will be limited to the set of nodes listed. Otherwise, the search will be performed for all nodes defined within the current management scope as determined by the CT_MANAGEMENT_SCOPE environment variable.

It is advisable to first use the **lsaudrec** command with the same **-s** and **-n** flag values to list the records that will be deleted. This minimizes the possibility of the selection string matching more records than intended.

Flags

-a Specifies that records from all nodes in the domain are to be removed. If both the **-n** and the **-a** flags are omitted, records from the local node only are removed.

-n *node_name1***[,***node_name2***]...**
Specifies the list of nodes containing audit log records that will be examined and considered for deletion 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 will be deleted.

-S *subsystem_name*
Specifies a subsystem name. If this flag is present, only records identified by *subsystem_name* are considered for deletion. The records to be deleted 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. If the evaluation results in a non-zero result (**TRUE**), the record is removed from the audit log. If the selection string contains any spaces, it must be enclosed within single or double quotation marks. For information on how to specify selection strings, see *RSCT: Administration Guide*.

The names of fields within 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 not specified, no records will be removed from the audit log.

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

-V Writes the command's verbose messages to standard error.

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 deleted. If the **-a** and **-n** flags are not specified, local scope is used. When either of these flags is specified, CT_MANAGEMENT_SCOPE is used to determine the management scope directly. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

If the **-V** flag is specified and the command completes successfully, a message indicating the number of records that were deleted will be 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.

Security

In order to remove records from an audit log when the **-S** flag is omitted, you must have write access to the target resource class on each node from which records are to be removed. When the **-S** flag is specified, you must have write access to the audit log resource corresponding to the subsystem identified by the **-S** flag on each node from which records are to be removed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmaudrec

Examples

1. To remove all records from the audit log on every node in the management scope defined by the CT_MANAGEMENT_SCOPE environment variable, enter:

   ```
   rmaudrec -s "Time > 0"
   ```

   or

   ```
   rmaudrec -s "SequenceNumber >= 0"
   ```

2. To remove all records more than a week old on every node in the management scope defined by the CT_MANAGEMENT_SCOPE environment variable, enter:

   ```
   rmaudrec -s "Time < #-0007"
   ```

3. To remove all records that are more than a day old and created by the **abc** subsystem on nodes **mynode** and **yournode**, enter:

   ```
   rmaudrec -S abc -s "Time < #-0001" -n mynode,yournode
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- how to specify selection strings
- using constants in expressions

Commands: **lsaudrec**

Part 6. Cluster security

Chapter 12. Cluster security services commands 305
ctaclfck 306
ctcasd 309
ctmsskf 311
ctscachgen 315
ctscfg 318
ctsidmck 322
ctskeygen 326
ctsthl 329
ctstrtcasd 333
ctsvhbac 335
ctsvhbal 339
ctsvhbar 342

© Copyright IBM Corp. 2002, 2007

Chapter 12. Cluster security services commands

© Copyright IBM Corp. 2002, 2007

ctaclfck

Purpose

Verifies the contents of a cluster security services ACL file.

Syntax

ctaclfck -f *acl_file_name* [**-s**] [**-c**] [**-u** *user_name*] [**-v**] [**-h**]

Description

The **ctaclfck** command checks the contents of the cluster security services ACL file specified by the **-f** flag. The check is limited to syntactical errors; a semantic check is not performed.

The command opens the ACL file, and reads and compiles one ACL entry at a time. If the command encounters an error, it will report the error to standard output. If the **-c** flag is provided, the command will continue processing after encountering errors until it reaches the end of the file. Otherwise processing will stop after the first error is found and reported.

The **-u** flag directs the command to verify the ACL file contents owned by the specified operating system's user identity. The command user must have permission to change to the home directory of the user specified by the **-u** flag, and must also have permission to read files in that directory. If the **-s** flag is specified along with the **-u** flag, the command user must also have permission to set its effective user identity to this identity (see the man page for the operating system command **su** for examples).

When the **-u** flag is specified, the file name provided in the **-f** flag is expected to be the base name of a file that resides in the home directory of the named user. In this case, the file name specified by the **-f** flag must not contain any directory names, including the **./** and **../** directories.

If the **-s** flag is specified, the command creates a file to contain the compiled contents of the ACL file. This permits applications to compile the ACL data buffer in advance to starting the application that uses it, saving the application this processing during its startup procedure or its ACL reading process. The compiled ACL file will have the same name as the ACL file with the extension **.cacl**. The ownership and file system permissions of the new ***.cacl** file will be set to the same ownership and permissions as the ACL file. If the ACL file is not currently owned by the command user, the command user must be capable of changing its effective user identity to the identity of the user that owns the ACL file. If the command is unable to do this, it will not create the ACL buffer file, but will complete verification of the ACL file.

The command checks for the correct ACL entry type, for the proper identity format, and for a valid permission. A valid permission is defined as one containing only operations that are defined by the permission template. The permission template set defined by cluster security services and used by this command follows.

r	0x1	read	generic read operation
w	0x2	write	generic write operation
c	0x4	control	generic control operation / RMC refresh configuration operation
x	0x8	run	generic execute operation
C	0x10	cancel	generic cancel operation
q	0x20	query	RMC query resource operation
l	0x40	list	RMC enumerated resources operation
e	0x80	event	RMC event registration / unregistration / querying
d	0x100	define	RMC define / undefine resource operation
v	0x200	validate	RMC validate resource handle operation
s	0x400	set	RMC set attribute operation

If the **-u** flag is specified, the command searches for the ACL file in the home directory of the specified user. The user must own the file and the permission must be write-only by the user. When the **-u** flag is specified, the ACL file name specified by the **-f** flag must not contain a relative or full path to the file; it must specify the file name only.

Flags

-f *acl_file_name*
Specifies the cluster security services ACL file to be verified. The file name can be a full or relative path name, unless the **-u** flag is specified.

-s Caches the ACL buffer (that resulted from the compilation of the ACL file) into a file. If the ACL file is not owned by the command user, the command user must be able to set its effective user identity to the owner of the ACL file.

–c Instructs the command to continue after encountering errors until the end of file is reached. All errors encountered will be reported regardless of whether or not the **-v** flag is specified. If not specified, command processing will stop after the first error is encountered and reported.

-u *user_name*
Specifies the user name in whose home directory the ACL file resides. When this flag is used, the file name specified by the **-f** flag must be the base name of a file that resides in the named user's home directory; the file cannot contain any directory information, including the **./** and **../** directory names.

-v Writes the command's verbose messages to standard output.

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

Exit status

0 The command completed successfully.

1 The command was able to verify the ACL file contents, but was unable to store the compiled ACL buffer into a file.

4 The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. This command terminated without processing the request.

6 A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

30 The command was unable to obtain exclusive use of the ACL file. Another instance of this command or another application may be running and accessing the ACL file. Retry the command at a later time.

36 The command user does not have permission to access the ACL file, or to read the contents of the ACL file.

105 The ACL file could not be located, or the contents of the file are corrupted.

Security

The file system permission of the ACL file is determined by the end user or the application that owns the file. If the invoker does not have sufficient authority to read the file or to create the requested compiled ACL file with the same ownership, the command fails.

Restrictions

The **ctaclfck** command works only on ACL files formatted for cluster security services.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctaclfck

Examples

1. To verify the contents of the ACL file **/my_acl_file**:

   ```
   ctaclfck -f /my_acl_file
   ```

2. To verify the contents of the ACL file **../my_acl_file** (relative to the current directory) and provide detailed output:

   ```
   ctaclfck -f ../my_acl_file -v
   ```

3. To completely verify the contents of the ACL file **/u/fluffy/my_acl_file**, which is owned by the operating system user **fluffy**, and store the compiled ACL buffer into a file for later use:

   ```
   ctaclfck -c -u fluffy -f  my_acl_file  -v -s
   ```

Related information

Commands: **su**

ctcasd

Purpose

Provides and authenticates the credentials of the RSCT host-based authentication (HBA) and enhanced host-based authentication (HBA2) security mechanisms for the cluster security services.

Syntax

ctcasd [**-b**]

Description

The **ctcasd** daemon is used by the cluster security services library when the RSCT HBA or HBA2 security mechanism is configured and active within the cluster environment. The cluster security services use **ctcasd** when service requesters and service providers try to create a secured execution environment.

When a service requester and a service provider have agreed to use the RSCT HBA or HBA2 mechanism through the cluster security services, the cluster security services library uses **ctcasd** to obtain and authenticate the RSCT HBA or HBA2 credentials. The cluster security services do not provide a direct interface to the daemon that can be invoked by user applications.

The **ctcasd** daemon can be started or stopped using system resource controller (SRC) commands.

During startup, the daemon obtains its operational parameters from the **ctcasd.cfg** configuration file. The daemon expects to find this file in the **/var/ct/cfg/** directory. System administrators can modify the operational parameters in this file to suit their needs. If this file is not located, the daemon will use the default configuration stored in **/usr/sbin/rsct/cfg/ctcasd.cfg**.

RSCT HBA and HBA2 credentials are derived from the local node's private and public keys. These keys are located in files that are configured in **ctcasd.cfg**. These credentials are encrypted using the public key of the receiving node. Public keys for the nodes within the cluster are stored in a trusted host list file on each node. The location of this file is also defined in the **ctcasd.cfg** configuration file. The system administrator is responsible for creating and maintaining this trusted host list, as well as for synchronizing the lists throughout the cluster.

If the daemon detects that the node's public key file and private key file are not present, **ctcasd** assumes that it is being started for the first time and thus creates these files. The daemon also creates the initial trusted host list file for this node. This file contains an entry for **localhost**, as well as the host names and IP addresses associated with all of the active, IPv4- and IPv6-configured adapters that the daemon can detect. Inadvertent authentication failures could occur if the public key file and the private key file were accidentally or intentionally removed from the local node before the daemon was restarted. **ctcasd** creates new keys for the node that do not match the keys stored on the other cluster nodes. If RSCT HBA or HBA2 suddenly fails after a system restart, this is a possible source of the failure.

Critical failures detected by the daemon that cause shutdown of the daemon are recorded to persistent storage. In AIX-based clusters, records are created in the AIX error log and the system log.

Flags

–b Starts the daemon in bootstrap mode. The daemon runs as a foreground process and is not controlled by the system resource controller (SRC).

Files

/usr/sbin/rsct/cfg/ctcasd.cfg	Default configuration for the **ctcasd** daemon
/var/ct/cfg/ctcasd.cfg	Configuration for the **ctcasd** daemon, which can be modified by the system administrator
/var/ct/cfg/ct_has.pkf	Default location of the cluster security services public key file for the node
/var/ct/cfg/ct_has.qkf	Default location of the cluster security services private key file for the node
/var/ct/cfg/ct_has.thl	Default location of the cluster security services trusted host list for the node

Restrictions

- The **ctcasd** daemon does not encrypt the RSCT HBA or HBA2 identity credentials.
- Cluster security services supports its own file formats, private key formats, and public key formats only. Cluster security services does not support secured remote shell formats.

Implementation specifics

This daemon is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctcasd

Related information

Commands: **ctskeygen**, **ctstrtcasd**, **ctsvhbac**, **ctsvhbal**, **ctsvhbar**, **ctsthl**, **startsrc**, **stopsrc**

Files: **ctcasd.cfg**, **ct_has.pkf**, **ct_has.qkf**, **ct_has.thl**

ctmsskf

Purpose

Displays and manages the contents of a message security services (MSS) key file.

Syntax

ctmsskf {**-a** | **-d** | **-l** | **-h**} [**-f** *key_file*] [**-t** *key_type*] [**-v** *key_version*] [**-k** *key_value*]

Description

The **ctmsskf** command displays and manages the contents of a message security services (MSS) typed key file. Use this command to add a key to, delete a key from, or list the contents of a key file.

Adding a key:

When you use this command to add a key entry to a key file, you must specify the following:

- the name of the key file where the key is to be added
- the type of the key to add
- optionally, the version of the key that is to be added to the key file
- the 16-digit value of the key

If the specified key file does not exist, it is created. If the specified key file *does* exist, the **ctmsskf** command verifies that the key type specified for the new key matches the type used by the keys already recorded within the file. Only keys of the same type can be added to an existing key file. When a key is successfully added to the file, that version of the key becomes the *active key version.* If a key version is specified using the **-v** *key_version* flag, *key_version* is used as the new version number and is made the active version. If *key_version* is not specified, the key is added using a key version value that is one greater than the previous active key version number.

Existing versions of a key cannot be replaced. To replace an existing version of a key or to change the value of an existing version of a key, that key version must first be deleted using the **-d** flag, and then added again using the **-a** flag. The command returns an error if you try to add a key that uses a version number already in use by a key within an existing key file. In general, key replacements should only be performed on the value of the key that is currently active, as replacing the value of an older key version makes the older key version active.

Because key versions can be added to the key file in any order, the highest key version number may or may not be the key version that is currently active. Use the **-l** flag to determine which key version is currently active for a file.

Deleting a key:

When you use this command to delete a key entry from a key file, you must specify the following:

- the name of the key file from where the key is to be deleted
- optionally, the type of key to delete
- optionally, the version of the key to delete

If the key specified is empty, does not exist, or does not have a proper header, the command returns an error. If the key type is specified and it does not match the key type in the header of the, the command returns an error. If the key version is specified, the command locates the record corresponding to the version provided and purges it from the file. If there is no such record, the command returns an error. If no key version is provided, the command purges only the records that are marked as inactive.

Listing the contents of a key file:

When you use this command to list the contents of a key file, the following information is displayed:

- the header of the key file.
- the list of keys in the key file.

  The following information is displayed for each key:
  - an indication of whether the record is inactive
  - the version of the key
  - the type of the key
  - the 16-digit value of the key

Flags

–a Adds a key to the key file. The **-f**, **-k**, and **-t** flags must also be specified.

–d Deletes a key from the key file. The **-f** and **-v** flags must also be specified. If the **-t** flag is specified, the command checks to see if the type of the key file is the same as the key type provided.

–l Lists the contents of the key file. The **-f** flag must also be specified. If the **-v** flag is specified, the command lists only the key that matches the version number provided.

–f *key_file*
Specifies the name of the key file. The key file must be a valid key file created by MSS API or by this command.

–t *key_type*
Specifies the type of the key to add. If the specified key file is not empty, the command checks to see if the key type specified matches the key type in the header of the key file. The valid key type values are: **3des_md5**, **aes256_md5**, **des_cbc**, **des_md5**, **rsa512_sha**, and **rsa1024_sha**.

–v *key_version*
Specifies the version of the key.

–k *key_value*
Specifies the 16-digit value of the key.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-i** flag is specified, the list of available key generation methods is displayed. When the **-l** flag is specified, one or more keys from the key file are displayed.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0 The command completed successfully.

4 The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. This command terminated without processing the request.

6 A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

9 If the **-a** flag was specified, the command detected a key within the key file that used the same version number as the one specified by the **-v** flag. If the **-d** flag was specified, the command was unable to locate a key in the key file using the version number specified by the **-v** flag. The key file was not modified.

21 The key file could not be located. Verify that the path name for the key file specified by the **-f** flag is correct.

27 The key type specified by the **-t** flag does not match the type for keys stored in the file specified by the **-f** flag. The requested action was not performed.

30 **ctmsskf** was unable to obtain exclusive use of the key file. Another instance of this command may be running and attempting to modify the same file, or the process that makes use of this key file may be examining the file. Retry the command at a later time.

36 The command user does not have sufficient permission to modify the contents of the key file.

37 The key file appears to be corrupted. Try to list the contents of the file using the **-l** flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Security

The file system permission of the key files is determined by the application owning the file. If the invoker doesn't have sufficient authority to open the file, the command fails.

Restrictions

This command works only on MSS-formatted key files.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctmsskf

Examples

1. To view the keys contained in the key file **/my_key_file**, enter:

```
ctmsskf -l -f /my_key_file
```

2. To view the key with version 9 from the key file **/my_key_file**, enter:
   ```
   ctmsskf -l -v 9 -f /my_key_file
   ```
3. To add a key to the key file **/my_key_file**, enter:
   ```
   ctmsskf -a -t des_cbc -f /my_key_file -k 16_digit_value
   ```
4. To delete a key from the key file **/my_key_file**, enter:
   ```
   ctmsskf -d -f /my_key_file -v 10
   ```
5. To delete all inactive keys in the key file **/my_key_file**, enter:
   ```
   ctmsskf -d -f /my_key_file
   ```

ctscachgen

Purpose

Creates or replaces an on-disk version of a key cache.

Syntax

ctscachgen -c *file-name* [**-f**] [**-i** | **-n** *enc-key-name* | **-k** *enc-key-value* **-t** *key-type* | **-q**] [**-m** *key-gen-method*] [**-s** *cache-size*] [**-h**]

Description

The **ctscachgen** command generates a key cache and stores the completed cache to an on-disk file named in *file-name*. This file can later be used and updated by applications through the **libct_skc** library interfaces.

Flags allow you to specify the type of key to be generated, using the mnemonics that are used for symmetric key types by the **ctmsskf** command. You can also specify a key value to be used to encrypt the keys available in this cache. The keys are not encrypted by default. In addition, you can specify the number of keys to be stored in the file.

If the file specified in *file-name* exists, it is overwritten, even if the current contents do not match the flags specified on the command line.

Flags

-c *file-name*
: Specifies the name of the key cache file. It can be either the full path or the relative path to the current directory.

-f
: Instructs the command to overwrite an existing key cache file with the same name without asking the invoker to confirm its overwriting.

-i
: Displays information about the key cache file specified with the **-c** flag. The information displayed contains the version of the cache file, the read count, the number of keys in the cache, the type of keys in the cache, and whether they are encrypted with a pre-encryption key. This flag cannot be used in conjunction with the **-n**, **-k**, **-t**, or **-q** flag.

-n *enc-key-name*
: Provides the name of the file that contains the encryption typed key. This flag cannot be used in conjunction with the **-i**, **-k**, **-t**, or **-q** flag.

-k *enc-key-value*
: Specifies the key value, expressed in hexadecimal form (**6fe45d20a**, for example), to be used as the pre-encryption key. By default, no pre-encryption key value is used. This flag must be used with the **-t** flag. It cannot be used in conjunction with the **-i**, **-n**, or **-q** flag.

-t *key-type*
: Provides the type of the encryption key specified by the **-k** option. The valid key types are: **3des_md5**, **aes256_md5**, **des_cbc**, **des_md5**, **rsa512_sha**, and **rsa1024_sha**. This flag must be used with the **-k** flag. It cannot be used in conjunction with the **-i**, **-n**, or **-q** flag.

–q
: Instructs the command to use the host's RSCT HBA or HBA2 private key as

encryption key used for pre-encrypting the session keys in the on-disk key cache file. This flag cannot be used in conjunction with the **-i**, **-k**, **-t**, or **-n** flag.

-m *key-gen-method*
Provides the session key generation method. Valid values are: **3des_md5**, **aes256_md5**, and **des_md5**. If you do not specify this flag, the default method for generating the session keys is **des_md5**.

-s *cache-size*
Provides the size of the on-disk key cache file in terms of number of keys in the cache. If you do not specify this flag, the default cache size is 128 keys.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-i** flag is specified, information about the key cache file is written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

Upon successful completion, the command returns an exit status code of **0** and generates an on-disk key cache file. In the event of a failure, the routine returns the error code and may remove the existing key cache file that the invoker wants to overwrite.

0 The command completed successfully.

4 Flags are mismatched or not valid. *file-name* remains unmodified.

6 A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

12 The command user cannot remove the existing key cache file (*file-name* remains unmodified) or access or write to the directory where *file-name* resides.

21 There is not enough space to store *file-name* or the *file-name* contents appear corrupt.

27 The key stored in the file specified by the **-c** flag is not valid or is corrupted. *file-name* remains unmodified.

36 The invoker cannot access the file specified by the **-c** flag. *file-name* remains unmodified.

Security

Permissions on the **ctscachgen** command permit only **root** to run the command.

Restrictions

- On-disk key caches are intended to be used solely upon the system on which they were generated. They are not intended to be shared between systems or migrated to another system. If multiple systems access the same key cache file, the protection offered by these keys is lost, because multiple systems and applications have access to information that is supposed to remain secret to a

specific application. Therefore, any files created by this command should not be stored in shared file systems or networked file systems.

- Files are generated by this command in a host-ordered binary format. This format makes it impossible for a key cache file generated on one architecture (such as a POWER™ platform) to be used on a different architecture (such as an Intel® platform).

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX .

Location

/usr/sbin/rsct/bin/ctscachgen

Related information

Commands: **ctmsskf**

ctscfg

Purpose

Lists and modifies the contents of the cluster security services configuration file.

Syntax

ctscfg -a { **-c** *MPM_code* } { **-n** *MPM_name* } { **-o** *MPM_object_module* } { **-p** *MPM_priority* } [**-f i** | **u** | **z**] [**-l**] [**-h**]

Syntax

ctscfg -d { **-c** *MPM_code* | **-n** *MPM_name* } [**-l**] [**-h**]

ctscfg -u { { **-c** *MPM_code* } | { **-n** *MPM_name* } } { { **-f i** | **u** | **z** } | { **-p** *MPM_priority* } } [**-l**] [**-h**]

ctscfg -l

ctscfg -h

Description

The **ctscfg** command lists and modifies the contents of the cluster security services configuration file, **ctsec.cfg**. This file provides configuration information about the authentication methods that cluster security services can use for client-server authentication. Each authentication method is handled by a mechanism pluggable module (MPM). Each MPM's configuration is defined by a one-line entry in the **ctsec.cfg** file. The entry contains information about:

- the priority of the MPM when cluster security services chooses the authentication method for the client-server authentication
- the numeric code of the MPM, which is unique among all of the MPMs in the configuration file
- the mnemonic of the MPM, which is unique among all of the MPMs in the configuration file
- the name of the binary module that implements the functionality of the MPM
- miscellaneous flags used by cluster security services' mechanism abstract layer (MAL) when handling the MPM

Cluster security services include a default **ctsec.cfg** file in the **/usr/sbin/rsct/cfg/** directory. The **ctscfg** command does not modify this default configuration file. Instead, **ctscfg** makes a copy (if one does not exist already) of the default **ctsec.cfg** file and copies it to the **/var/ct/cfg/** directory. If a working copy of this file does exist already and there is enough space, the previous version is recorded to **/var/ct/cfg/ctsec.cfg.bak**.

Using this command, system administrators can create an ″empty″ security subsystem configuration, where no security MPMs are configured. In this configuration, all parties are are to be considered not authentic.

Flags

-a Adds a new configuration entry for a new MPM to the working copy of the **ctsec.cfg** file in the **/var/ct/cfg/** directory. If there is no working copy in that directory, **ctscfg** creates a working copy and modifies it. A configuration

entry must include the MPM priority, numeric code, mnemonic, binary object, and, optionally, any flags. This flag requires the **-c**, **-n**, **-o**, and **-p** flags.

-c *MPM_code*
Specifies the code to be used by the security subsystem to refer to this MPM. *MPM_code* must be expressed as a hexadecimal value in the form of ″**0x***value*″ (″**0x1a**″ or ″**0x9F**″, for example). This flag is required by the **-a** and **-d** flags.

-d Removes an existing entry for a security MPM from the working copy of the **ctsec.cfg** file in **/var/ct/cfg**. If there is no working copy in that directory, **ctscfg** creates a working copy and modifies it. The **-c** flag or the **-n** flag must be specified to indicate which entry is to be removed.

-f i | **u** | **z**
Specifies the flags required by the security subsystem when adding an MPM to the configuration file. This option is required by the **-a** flag if the MPM has any miscellaneous flags or by the **-u** flag if the invoker intends to update the MPM flags. The MAL supports these miscellaneous flags:

i instructs MAL to initialize the MPM upon loading it in the process's virtual memory.

u instructs MAL that it is safe to unload the MPM when it is no longer required.

z specifies the authorization method used for that MPM. An MPM with the same mnemonic as the authorization method must also exist and be configured in **ctsec.cfg**.

The flags must be specified with no space between them (**-f iuz**, for example).

-l Lists the contents of the working **ctsec.cfg** file. If this option is specified with **-a**, **-d**, or **-u**, the resulting configuration is listed.

-n *MPM_name*
Specifies the mnemonic to be used for the security MPM. The mnemonic should be a short string value (**mymech**, for example). This flag is required by the **-a** and **-d** flags.

-o *MPM_object_module*
Specifies the location of the MPM, including the full path subdirectory. The MPM must exist as a file. If a symbolic link is used, the symbolic link must reference an existing file. The path must be expressed as an absolute path (**/usr/lib/mymech**, for example). This flag is required by the **-a** flag.

-p *MPM_priority*
Specifies the priority associated with this security mechanism pluggable module (MPM). Lower values have a higher priority. Priority values do not need to be consecutive, but no two MPMs can share the same priority. Negative values and a zero value are not permitted for a priority. This option is required by the **-a** flag and the **-u** flag if the invoker intends to update the MPM priority.

–u Updates an existing configuration entry of an MPM in the working copy of the **ctsec.cfg** file in **/var/ct/cfg**. If there is no working copy in that directory, **ctscfg** creates a working copy and modifies it. The configuration entry must be specified by either the MPM numeric code or mnemonic. The only fields that can be updated are the MPM priority and flags. This flag requires the

-c flag or the **-n** flag (in order to identify the configuration entry to modify) and **-f** flag or the **-p** flag (to specify the new values used for updating the selected configuration entry).

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0 The command completed successfully.

4 Flag error. One or more of the flags provided is not valid or is missing a value.

21 Configuration error. The MAL configuration file's content is not valid or is corrupted.

30 Lock error. An error occurred during the locking of the MAL's configuration file.

36 Permission error. The invoker does not have permission to list or modify the MAL's configuration file.

105 File error. An error occurred during the reading or writing of the MAL's configuration file.

Files

/var/ct/cfg/ctsec.cfg	Working copy of the MAL's configuration file
/var/ct/cfg/ctsec.cfg.bak	Backup of the working copy of the MAL's configuration file

Security

This command lists and modifies the MAL's configuration file. The default version of the MAL configuration file that is installed by RSCT is protected using the file system's permission bit mask of 444 (that is, read-only for everybody). Administrators who create a working copy of this file must preserve the permission bit mask in order to maintain the security of the system.

This command uses the working copy of the MAL's configuration file in **/var/ct/cfg/**. If there is no such working copy, the command creates a file with the same ownership and permission bit mask as the default configuration file. If the invoker of the command has no permission to do that, the command returns a permission error.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX .

Location

/usr/sbin/rsct/bin/ctscfg

Examples

1. To list the contents of the working copy of the **ctsec.cfg** file, either in **/usr/sbin/rsct/cfg/** or in **/var/ct/cfg/**, enter:

   ```
   /usr/sbin/rsct/bin/ctscfg -l
   ```

2. To add the HBA2 MPM to the working copy of the **ctsec.cfg** file in **/var/ct/cfg/**, enter:

   ```
   /usr/sbin/rsct/bin/ctscfg -a -n hba2 -p 2 -c 0x2 -o /usr/sbin/rsct/lib/hba2.mpm -f i
   ```

   This adds the following record to the working copy of the **ctsec.cfg** file in **/var/ct/cfg/**:

   ```
   1      hba2        0x00002     /usr/lib/hba2.mpm       i
   ```

3. To delete the UNIX MPM from the working copy of the ctsec.cfg file in /var/ct/cfg/, enter:

   ```
   /usr/sbin/rsct/bin/ctscfg -d -n unix
   ```

4. To update the HBA2 MPM with the UNIX MPM as the new authorization method in the working copy of the **ctsec.cfg** file in **/var/ct/cfg/**, enter:

   ```
   /usr/sbin/rsct/bin/ctscfg -u -n hba2 -f iz [unix]
   ```

5. To update the priority of the HBA2 MPM to a value of 2 in the working copy of the **ctsec.cfg** file in **/var/ct/cfg/**, enter:

   ```
   /usr/sbin/rsct/bin/ctscfg -u -n hba2 -p 2
   ```

Related information

Files: **ctsec.cfg**

ctsidmck

Purpose

Verifies the cluster security library identity mapping.

Syntax

ctsidmck -h | **-i** | { [**-dl** | **-dm** | **-dh**] **-m** *security_mechanism network_ID* }

Description

A system administrator can use the **ctsidmck** command to verify the mapping that would be obtained by the cluster security library (**libct_sec**) for a specific security network identifier.

The cluster security library establishes a security context through the exchange between a client of a trusted service and the trusted service's server. During the creation of the security context, the cluster security library tries to map the client application's security network identity to an identity that may be present on the server node, called the *mapped identity.* The cluster security library uses the mapped identity later on the server in such authorization functions as access control verification. Whether the client application has a mapped identity on the server depends on whether the following identity mapping definition files are present on the server, and whether any of the entries within these files correspond to the security identity being used by the client application:

- **/usr/sbin/rsct/cfg/ctsec_map.global**
- **/var/ct/cfg/ctsec_map.local**
- **/var/ct/cfg/ctsec_map.global**

The location of definitions within these files is important; entries at the head of the file are processed before entries positioned towards the end of the file. The definition rules also allow for wildcarding of entry information and for expansion of certain reserved words. If a definition is incorrectly specified within one of these files, the mapping result may not be as intended. Also, if a definition is positioned after another definition that can successfully map a security network identifier, the mapping result may not be as intended.

Using **ctsidmck**, an administrator can verify that the correct identity mapping definition is used by the cluster security library to map a security network identity. This command is to be executed on the node that would act as the server. By specifying a security network identifier to this command on the server, the administrator can determine what the mapped identity for that security network identity would be on that node, and what entry was used from the identity mapping definition files to obtain this mapping.

Parameters

network_ID
: Specifies the security network identifier to be mapped. This should be an identity that can be assumed by a client application of a trusted service.

Flags

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

–i Displays a list of the supported security mechanisms on this node. The

command examines the cluster security library configuration on this node, obtains a list of supported security mechanisms, and displays this list. The mechanisms are listed by the mnemonic used by the cluster security library to refer to these mechanisms.

–d Specifies the level of detail in the command output. One of three levels of detail is permitted:

1. low (**l**): the command will only display the mapped identity for *network_ID*. This is the default detail level.
2. medium (**m**): the command will display the mapped identity for *network_ID*, as well as the entry from the identity mapping definition files that yielded the map.
3. high (**h**): the command will display every entry from the identity mapping definition files that is processed until a mapped identity for *network_ID* is found, or until all entries are processed.

-m *security_mechanism*

Specifies the security mechanism that was used to create the security network identifier provided by *network_ID*. *security_mechanism* is a mnemonic that would be used by the cluster security library to refer to this security mechanism. This flag must be specified when the **-h** and the **-i** flags are not provided.

Use the **-i** flag to display a list of the security mechanisms that this node supports.

Files

/usr/sbin/rsct/cfg/ctsec_map.global
The default identity mapping definition file. This file contains definitions required by the RSCT cluster trusted services in order for these systems to execute properly immediately after software installation. This file is ignored if the cluster-wide identity mapping definition file **/var/ct/cfg/ctsec_map.global** exists on the system. Therefore, any definitions within this file should also be included in the cluster-wide identity mapping definition file, if that file exists.

/var/ct/cfg/ctsec_map.local Local override to the cluster-wide identity mapping definitions. Definitions within this file are not expected to be shared between nodes within the cluster.

/var/ct/cfg/ctsec_map.global Cluster-wide identity mapping definitions. This file is expected to contain identity mapping definitions that are common throughout the cluster. If this file exists on the system, the default identity mapping definition file is ignored. Therefore, if this file exists, it should also contain any entries that would also be found in the default identity mapping definition file.

Standard output

The **ctsidmck** command writes any mapped identity found for the security network identifier to standard output. If a medium or high level of detail is requested, any definitions displayed by this command are also written to standard output.

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0 This command successfully found a mapped identity for *network_ID*.

3 This command detected a failure in the operation of the cluster security library mechanism pluggable module (MPM) corresponding to the security mechanism that was requested. **ctsidmck** was unable to search for a possible mapped identity for *network_ID* in this case. This failure may be accompanied by descriptive output indicating the nature of the MPM failure. Consult this output and perform any recommended actions.

4 The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. **ctsidmck** terminated without trying to find a mapped identity for *network_ID*.

6 A memory allocation request failed during the operation of this command. **ctsidmck** was unable to search for a possible mapped identity for *network_ID* in this case.

21 This command was unable to locate any of the identity mapping definition files on the local node. **ctsidmck** was unable to search for a possible mapped identity for *network_ID* in this case. Verify that at least one identity mapping definition file exists on the system.

22 This command was unable to dynamically load the cluster security library mechanism pluggable module (MPM) corresponding to the security mechanism what was requested. The module may be missing, corrupted, or one of the shared libraries used by this module may be missing or corrupted. **ctsidmck** was unable to search for a possible mapped identity for *network_ID* in this case. This failure may be accompanied by descriptive output indicating the nature of the MPM failure. Consult this output and perform any recommended actions.

37 At least one of the identity mapping definition files on the system appears to be corrupted. The command was unable to search for a possible mapped identity for *network_ID* in this case. Verify that none of the identity mapping files are corrupted, truncated, or contain syntax errors.

38 ctsidmck could not locate a mapped identity for *network_ID*. No entry within any of the identity mapping definition files yielded a mapped identity for the specified security network identifier.

Security

This command is executable only by the root system user and members of the system user group. It is intended for administrator use only, to verify the security configuration of the system. Because the output of the command could be used as a means for determining how to sabotage or circumvent system security, the permissions on this command should not be altered.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsidmck

Examples

1. To get a list of the security mechanisms that the local node supports, before verifying an identity map, enter:

   ```
   ctsidmck -i
   ```

2. To get only the mapped identity for the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism's network identity **zathras@greatmachine.epsilon3.org**, enter:

   ```
   ctsidmck -m unix zathras@greatmachine.epsilon3.org
   ```

3. To see every identity mapping definition that the command checks while searching for a mapped identity for the RSCT HBA or HBA2 security mechanism's network identity **glorfindel@rivendell.elvin.net@endor**, enter:

   ```
   ctsidmck -d h -m unix glorfindel@rivendell.elvin.net@endor
   ```

Related information

Files: **ctsec_map.global**, **ctsec_map.global**

ctskeygen

Purpose

Generates cluster security services private and public keys for the local node and stores these keys in locally-mounted files.

Syntax

ctskeygen -n [**-f**] [**-m** *method*] [**-p** *public-file*] [**-q** *private-file*] | **-d** | **-i** | **-h**

Description

The **ctskeygen** command generates host identifier keys — a private key and public key pair — to be used by the cluster security services library (**libct_sec**) in RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2). The command creates a new private key for the node, derives a public key from the new private key, and stores these keys to files on the local node.

Whenever the node's private and public keys are modified, the node's new public key must be distributed to all nodes within the cluster and placed in the trusted host list files on these nodes, replacing the previous value stored there for this node. If this is not done, the node that has generated new private and public keys will be unable to authenticate with other nodes in the cluster using RSCT HBA or HBA2.

Flags

–n Generates host identifier keys (private and public keys).

–f Forces **ctskeygen** to record the keys it generates to the private and public key files if these files already exist. By default, the command will not overwrite these files if they exist, because the presence of the files indicates that the cluster security services service may be active. Removing or modifying these files without informing other nodes of the change in the public key value will cause failures in RSCT HBA or HBA2 on this node. This flag is not valid with the **-h** or the **-i** flag.

–m *method*
Instructs the command to use the specified key generation method in creating the host identifier keys. Valid parameters for this flag can be displayed using the **-i** flag. This flag is not valid with the **-h** and **-i** flags.

–p *public-file*
Specified the fully-qualified path name of the file to be used to store the local host's public key. If this file exists, the command will not overwrite the contents of this file unless the **-f** flag is also specified. If the **-p** flag is not specified, the command records this key to the **/var/ct/cfg/ct_has.pkf** file. This flag is not valid with the **-h** and **-i** flags.

–q private-file
Specified the fully qualified path name of the file to be used to store the private key of the local host. If this file exists, the command will not overwrite the contents of this file unless the **-f** flag is also specified. If the **-q** option is not specified, the command records this key to the file **/var/ct/cfg/ct_has.qkf**. This flag is not valid with the **-h** and **-i** flags.

–d Displays the current public key value for the local node.

–i Displays information about the key generation methods supported by this version of the command. **ctskeygen** displays messages to indicate which

values are currently supported as arguments to the **-m** flag, and what the command will use as a default setting for the **-m** flag.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-d** flag is specified, the public key value stored in the public key file is written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0 The command completed successfully.

4 The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually-exclusive flags. This command terminated without processing the request.

6 A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

12 The command user does not have sufficient permission to view or modify the contents of the key file.

21 The key file could not be located or could not be created.

30 **ctskeygen** was unable to obtain exclusive use of the public or private key file. Another instance of this command may be running and attempting to modify the keys, or the **ctcasd** daemon may be examining these files. Retry the command at a later time.

37 The public or private key file appears to be corrupted. Try to view the public key value using the **-d** flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Security

Permissions on the **ctskeygen** command permit only **root** to run the command.

Restrictions

- Cluster security services supports its own file formats, private key formats, and public key formats only.
- Trusted host lists are modifiable using the **ctsthl** command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctskeygen

Examples

1. To obtain the list of supported key generation methods:
   ```
   ctskeygen -i
   ```
2. To create new host identifier keys for the local node using the default settings:
   ```
   ctskeygen -n
   ```
3. To create new host identifier keys for the local node using 512-bit RSA private keys, storing these keys in locations other than the default location:
   ```
   ctskeygen -n -m rsa512 -p /mysec/public -q /mysec/private
   ```

Related information

Commands: **ctsthl**, **ctsvhbac**

Daemons: **ctcasd**

Files: **ct_has.pkf**, **ct_has.qkf**, **ct_has.thl**

ctsthl

Purpose

Displays and modifies the contents of a cluster security services trusted host list file.

Syntax

ctsthl {**-a** | **-d** | **-h** | **-l** | **-s** } [**-f** *trusted_host_list_file*] [**-n** *host_name*] [**-m** *method*] [**-p** *identifier_value*]

Description

This command displays and modifies the contents of a cluster security services trusted host list file. Unless the **-f** flag is provided, the command performs its operations on the trusted host list file configured in the **ctcasd.cfg** file. **ctsthl** allows the command user to add, modify, or remove entries in the trusted host list for specific hosts. When a host is added or modified, the command user must provide the following information:

- The identity of the host (**zathras.unixrulz.com** or **129.34.128.54**, for example)
- The host identifier value to be used for this host, in a character string format representing the identifier's hexadecimal value (**b87c55e0**, for example)
- The method that was used to generate the host identifier (see the description of the **ctskeygen -i** command)

The command validates the generation method name, converts the character string representation to binary form, and creates a new entry within the trusted host list file for this host. Generally, the host identifier value is quite large. For instance, the character representation of a RSA 1024-bit generated identifier is over 256 characters in size. This can cause a problem on operating systems such as AIX, which limit the command line length to a smaller size. To avoid this problem, use the **ctsthl -a** command from a shell script, or in conjunction with the **xargs** command.

When the contents of the trusted host list file are displayed, **ctsthl** provides the following information for each entry:

- The network identity of the host
- The host identifier value for that host, represented as a character string
- The method used to generate the host identifier

Flags

–a Adds to or replaces a host entry in the trusted host list. The **-n**, **-m**, and **-p** flags also must be provided. If the host specified already exists in the trusted host list file, the entry for that host is modified to match the information provided to this command.

–d Removes a host's entry from the trusted host list file. The **-n** flag also must be provided to indicate the host being removed.

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

–l Instructs the command to list the contents of the trusted host list file. If this flag is combined with the **-a** or **-d** flags, the contents are displayed after these flags are processed. If this flag is combined with the **-s** flag, any new

entries made by the command are displayed, as well as any public key mismatches detected for host names and IP addresses supported by the local node.

–f *trusted_host_list_file*
Specifies the fully-qualified path name of the trusted host list file. If this flag is not provided, the trusted host list file configured in the **ctcasd.cfg** file is used.

–n *host_name*
Specifies the identity of the host to be used in this operation. The identity should be a host name or IP address specification by which the host is known to the cluster's network.

–m *method*
Instructs the command to use the specified key generation method in creating the host identifier keys. You can use the **ctskeygen -i** command to display valid values for *method*.

–p *identifier_value*
Specifies the host identifier value to be stored for the host. This is a character string that represents the hexadecimal value of the host identifier to be stored for this identifier. For example, if the host identifier value is **0xB87C55E0**, this flag would be specified as **-p b87c55e0**. Generally, host identifier keys will be much longer than this example, making it too large for the command line limit on some operating systems, such as AIX. If the resulting command line is too large, use **xargs** to extend it, or issue the command from a shell script.

–s Explores the local node for all known IP addresses and host names associated with all of the active, IPv4- and IPv6-configured adapters that the daemon can detect. For any host name or IP address on the local node that is not found in the local node's trusted host list file, an entry is added to associate that value with the local node's public key value.

Files

/usr/sbin/rsct/cfg/ctcasd.cfg	Default configuration for the **ctcasd** daemon
/var/ct/cfg/ctcasd.cfg	Configuration for the **ctcasd** daemon, which can be modified by the system administrator

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. When the **-l** flag is specified, the contents of the trusted host list file are written to standard output.

Standard error

Descriptive information for any detected failure condition is written to standard error.

Exit status

0 The command completed successfully.

4 The caller invoked this command incorrectly, omitting required flags and parameters, or using mutually exclusive flags. This command terminated without processing the request.

6 A memory allocation request failed during the operation of this command. The command was unable to complete the requested action.

10 The command was unable to locate any configured and active network (IPv4 or IPv6) interfaces for the local node while processing the **-s** flag. The local node's identities may not be properly recorded to the trusted host list. Verify that at least one IPv4 or IPv6 interface is defined and active on the local node and reissue the command.

12 The command user does not have sufficient permission to view or modify the contents of the trusted host list file.

21 The trusted host list file could not be located, or could not be extended to contain a new public key value.

30 **ctsthl** was unable to obtain exclusive use of the trusted host list file. Another instance of this command may be running and attempting to modify the keys, or the **ctcasd** daemon may be examining these files. Retry the command at a later time.

31 The public key value specified by the **-p** flag does not end on a full byte boundary. Make sure the value contains an even number of digits.

37 The key file appears to be corrupted. Try to view the public key value using the **-d** flag to verify if the file is corrupted. Follow the problem resolution advice listed in the error message for further recovery action.

Security

Permissions on the **ctsthl** command permit only **root** to run the command.

Restrictions

- Cluster security services supports its own host identifier format and trusted host list file format only.
- Trusted host lists are modifiable using this command only.
- Cluster security services does not provide an automated utility for creating, managing, and maintaining trusted host lists throughout the cluster. This is a procedure left to either the system administrator or the cluster management software.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsthl

Examples

1. To view the contents of the trusted host contained in the file **/mythl**, enter:

   ```
   ctsthl -l -f /mythl
   ```

2. To add an entry to the default trusted host list file for the node **zathras.unixrulz.com**, enter:

   ```
   ctsthl -a -n zathras.ibm.com -m rsa1024 -p 120400a9...
   ```

   Note that this example does not complete the entire identifier value.

3. To add an entry to the default trusted host list file for the node **129.23.128.76**, enter:

   ```
   ctsthl -a -n 129.23.128.76 -m rsa1024 -p 120400a9...
   ```

   Note that this example does not complete the entire identifier value.
4. To remove an entry for zathras.ibm.com from the default trusted host list, enter:

   ```
   ctsthl -d -n zathras.ibm.com
   ```

Related information

Commands: **ctskeygen**

Daemons: **ctcasd**

Files: **ctcasd.cfg**

Books: see the cluster security topics in *RSCT: Administration Guide*

ctstrtcasd

Purpose

Serves as the **ctcasd** daemon's launch utility for the cluster security services.

Syntax

ctstrtcasd [**-a**] [**-v**]

Description

The **ctstrtcasd** utility is invoked by the cluster security services to start the **ctcasd** daemon. This utility is provided as a set-user-identity-on-execution binary file, giving cluster security services' clients the ability to start the **ctcasd** daemon through the system resource controller (SRC).

The **ctcasd** daemon is used by the cluster security services library when the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism is configured and active within the cluster environment. The cluster security services use **ctcasd** when service requesters and service providers try to create a secured execution environment.

When a service requester and a service provider have agreed to use the RSCT HBA or HBA2 mechanism through the cluster security services, the cluster security services library uses **ctcasd** to obtain and authenticate the RSCT HBA or HBA2 credentials. The cluster security services do not provide a direct interface to the daemon that can be invoked by user applications.

ctcasd is registered with the SRC as the **ctcas** subsystem. This subsystem is not activated by the SRC until the cluster security services receive a request for the RSCT HBA or HBA2 mechanism. SRC subsystems can only be activated by the system superuser. To allow the cluster security services to process HBA or HBA2 requests for any system user, the cluster security services must be able to activate the **ctcas** subsystem for normal system users as well as the system superuser if the service is not already active. To grant normal system users this ability, the cluster security services invoke the **ctstrtcasd** utility to start the **ctcas** subsystem if the service is not active. This utility temporarily grants the cluster security services' client user sufficient privilege to start the **ctcas** subsystem.

Flags

–a Verifies that the **ctcas** subsystem is operational and can process requests from the cluster security services after it is started.

–v Specifies that **ctstrtcasd** will display status information to standard output and and error information to standard error in verbose mode.

Standard output

When the **-v** flag is specified, this command's status information is written to standard output.

Standard error

When the **-v** flag is specified, this command's error information is written to standard error.

Security

The **ctstrtcasd** utility, a set-user-identity-on-execution binary file, is owned by the **root** system user. This special permission and ownership is required to temporarily grant cluster security services' clients the ability to start the **ctcas** subsystem if it is not already active on the system. Without this permission and ownership, some cluster security services' clients may not be able to start the **ctcasd** daemon to handle cluster security services requests, which can result in authentication failures.

See the ″Diagnosing cluster security services problems″ chapter of the *RSCT: Diagnosis Guide* for more information about the ownership and permissions required for this utility.

Restrictions

This utility is only intended for use by the cluster security services library or as directed by an IBM service representative.

Implementation specifics

This utility is part of the Reliable Scalable Cluster Technology (RSCT) cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctstrtcasd

Related information

Books: *RSCT: Diagnosis Guide*, for additional information about the ownership and permissions required for this utility

Commands: **ctcasd**, **startsrc**, **stopsrc**

ctsvhbac

Purpose

Verifies the configuration of the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism on the local node.

Syntax

ctsvhbac [[**-d** | **-h** | **-m** | **-s**] | [**-e** *msgnum*[,*msgnum*...]] [**-l** { **1** | **2** | **3** | **4** } | **-b**] [**-p** *public_key_file*] [**-q** *private_key_file*] [**-t** *thl_file*]]

Description

The **ctsvhbac** command is a verification utility for the RSCT HBA or HBA2 security mechanism. Use the **ctsvhbac** command to verify that the local node has configuration and credential files and information, such as private keys and a trusted host list, ready for the RSCT HBA or HBA2 security mechanism to use.

This command performs the following series of tests on the configuration of the RSCT HBA or HBA2 security mechanism:

- Verifies that the RSCT HBA or HBA2 mechanism configuration file is available and can be processed.
- Verifies that the RSCT HBA or HBA2 private key file exists and can be processed.
- Verifies that the RSCT HBA or HBA2 public key file exists and can be processed.
- Verifies that the private and public keys for the local node are in pairs, which means that the public key is known to be derived from the private key.
- Verifies that the RSCT HBA or HBA2 trusted host list file exists and can be processed.
- Checks the contents of the RSCT HBA or HBA2 trusted host list for all of the host names and network addresses supported by the local node, determining whether entries exist in the trusted host list file for them. If a host name or network address is found, the command verifies that the same public key value that was used in earlier tests is listed for the name or address.

You can specify the private key file, public key file, and trusted host list file to use in the command. By default, this information is extracted from the configuration file for the RSCT HBA or HBA2 security mechanism.

For more detailed information about configuration requirements for authentication, see the cluster security topics in *RSCT: Administration Guide*.

Parameters

None.

Flags

-b Produces brief output. When this option is used, the command displays only summary output of the tests and any errors detected. Further details of any errors can be determined by reissuing this command without this option. If the **-l** option is specified, this option is ignored.

-d Displays the list of probes required for successful execution of this command.

-e *msgnum*[**,***msgnum*...
Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the *nnnn-nnn* format. Multiple messages are to be separated by commas (**,**) with no blank spaces in the character string.

-h Displays a help message for this command.

-l Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

1 Verbose mode. Displays the command purpose summary and status information for all tests.

2 Displays the command purpose summary and any attention or error conditions detected in any tests.

3 Displays any attention or error conditions detected in any tests.

4 Silent mode. Displays errors detected during the tests.

-m Displays a detailed description of the command and its purpose.

-p *public_key_filethl_file*
Specifies the path name of the public key file that is to be used by the command. If this option is not specified, the command will use the public key file currently configured for the RSCT HBA or HBA2 security mechanism.

-q *private_key_file*
Specifies the path name of the private key file that is to be used by the command. If this option is not specified, the command will use the private key file currently configured for the RSCT HBA or HBA2 security mechanism.

-s Displays a summary of the purpose for the command.

-t *thl_file* Specifies the path name of the trusted host list file that is to be used by the command. If this option is not specified, the command will use the trusted host list file currently configured for the RSCT HBA or HBA2 security mechanism.

Exit status

Exit status conforms to the CSM Probe Infrastructure conventions.

0 No problems detected. Any messages displayed either are informational or indicate only minor alerts. No administration intervention is required.

10 No problems were detected, but some items found warrant administrator attention. This exit status most commonly occurs if an IP address or host name supported by the local node is not listed in the trusted host list, or is listed with an incorrect public key value. For this exit status, the system administrator should examine the output to determine which conditions were detected, and whether they require corrective action.

To correct the most commonly reported conditions:

- Ensure that any IP addresses or host names that are not in the trusted host list were purposely omitted. If not, update the trusted host list on the local node.
- Repair any entries for local IP addresses and host names that use incorrect public keys.

20 One or more problems were detected. This exit status occurs for the following conditions:
- The RSCT HBA or HBA2 security mechanism is configured incorrectly.
- Public and private keys might not be in pair.
- The trusted host list contains none of the IP address or host name values supported by the local node.

Unless these conditions are corrected, authentication requests using the RSCT HBA or HBA2 mechanism probably will not be successful on this system. For this exit status, the system administrator must examine the command output to identify and resolve reported problems. To correct reported problems, follow the problem-resolution advice listed in the command output.

127 Unexpected failure in this command. For this exit status, the administrator should verify that at least one network interface is both configured and active on this system.

Security

Permissions on the **ctsvhbac** command permit members of the **bin** user group to execute this command.

Implementation specifics

This command is part of the RSCT cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsvhbac

Examples

Suppose the local node is using the RSCT HBA security mechanism. To verify the RSCT HBA configuration, enter:

```
ctsvhbac
```

The output will look like this:

```
-------------------------------------------------------------------------
Host Based Authentication Mechanism Verification Check

Private and Public Key Verifications

      Configuration file:  /usr/sbin/rsct/cfg/ctcasd.cfg
                  Status:  Available
                Key Type:  rsa512
                           RSA key generation method, 512-bit key

        Private Key file:  /var/ct/cfg/ct_has.qkf
                  Source:  Configuration file
                  Status:  Available
                Key Type:  rsa512
                           RSA key generation method, 512-bit key
```

```
          Public Key file:  /var/ct/cfg/ct_has.pkf
                   Source:  Configuration file
                   Status:  Available
                 Key Type:  rsa512
                            RSA key generation method, 512-bit key

               Key Parity:  Public and private keys are in pair

Trusted Host List File Verifications

  Trusted Host List file:  /var/ct/cfg/ct_has.thl
                   Source:  Configuration file
                   Status:  Available

                 Identity:  avenger.unixrulz.com
                   Status:  Trusted host

                 Identity:  9.117.10.4
                   Status:  Trusted host

                 Identity:  localhost
                   Status:  Trusted host

                 Identity:  127.0.0.1
                   Status:  Trusted host

Host Based Authentication Mechanism Verification Check completed
```

Related information

Commands: **ctskeygen**, **ctsthl**, **ctsvhbal**, **ctsvhbar**

Files: **ctcasd.cfg**, **ct_has.pkf**, **ct_has.qkf**, **ct_has.thl**

Books: See the cluster security topics in *RSCT: Administration Guide*

ctsvhbal

Purpose

Displays the possible identities that the local node may use to identify itself in RSCT host-based authentication (HBA) and enhanced host-based authentication (HBA2) security mechanism credentials.

Syntax

ctsvhbal [[**-d** | **-h** | **-m** | **-s**] | [**-e** *msgnum*[,*msgnum*...]] [**-l** { **1** | **2** | **3** | **4** } | **-b**]

Description

The **ctsvhbal** command is a verification utility for the RSCT HBA and HBA2 security mechanisms. It displays the possible identities that the local node may use to identify itself in RSCT HBA or HBA2 credentials.

The RSCT HBA or HBA2 security mechanism might use either a host name or a network address value as part of the identification information within a credential, depending on the method chosen by the application. If the local node is to service requests from remote nodes, at least one network address and host name for that remote node must appear in the trusted host list on the local node. To verify that the remote node can successfully authenticate the local node, system administrators can use a combination of RSCT cluster security commands:

1. On the local node and the remote node, issue the **ctsvhbac** command to verify that each node has a valid RSCT HBA or HBA2 security mechanism configuration.
2. On the local node, issue the **ctsvhbal** command to determine the values that the RSCT HBA or HBA2 security mechanism will use to identify this host to a remote node.
3. On the remote node, issue the **ctsvhbar** command, specifying the local node's host name or IP address, to determine the value that the remote node will use to verify RSCT HBA or HBA2 credentials transmitted from the local node.
4. Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two nodes are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these steps verifies successful authentication in one direction; in other words, the procedure verifies only that the remote node can authenticate requests from the local node. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local node can successfully authenticate the remote node. To complete the verification, the following additional steps are required:

- On the remote node, issue the **ctsvhbal** command to determine the values that the RSCT HBA or HBA2 security mechanism will use to identify that host to the local node.
- On the local node, issue the **ctsvhbar** command, specifying the remote node's host name or IP address, to determine the value that the local node will use to verify RSCT HBA or HBA2 credentials transmitted from the remote node.
- Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two nodes are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these additional steps verifies successful authentication when traffic flows in the opposite direction, from the remote node to the local node.

For more detailed instructions and examples, see the cluster security topics in *RSCT: Administration Guide*.

Flags

-b Produces brief output. When this option is used, the command displays the host identities that are found for the local node and any errors that are detected. If the **-l** option is specified, this option is ignored.

-d Displays the list of probes required for successful execution of this command.

-e Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the *nnnnxxxx-yyy* format. Multiple messages are to be separated by commas (,) with no white space characters.

-h Displays a help message for this command.

-l Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

- **1** Verbose mode. Displays the command purpose summary and status information for all tests.
- **2** Displays the command purpose summary and any attention or error conditions detected in any tests.
- **3** Displays any attention or error conditions detected in any tests.
- **4** Silent mode. Displays errors detected during the tests.

-m Displays a detailed description of the command and its purpose.

-s Displays a summary of the purpose for the command.

Exit status

Exit status conforms to the CSM Probe Infrastructure conventions.

0 No problems detected. Any messages displayed are informational. No administration intervention is required.

10 No problems were detected, but the local node is unable to authenticate itself to any remote node. The local node does not have any active network interfaces, which is a configuration that RSCT permits. For this exit status, however, the system administrator should verify that this configuration is appropriate.

20 One or more problems were detected. Host-name resolution mechanisms that the local node uses are unable to obtain host names of network interfaces that the local node supports. Unless this condition is corrected, authentication requests using the RSCT HBA or HBA2 mechanism probably will not be successful on this node. For this exit status, the system administrator should follow the problem-resolution advice listed in the command output.

127 Unexpected failure in this command.

Security

Permissions on the **ctsvhbal** command permit members of the **bin** user group to execute this command.

Implementation specifics

This command is part of the RSCT cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsvhbal

Examples

Suppose the local node is using the RSCT HBA security mechanism. To display the possible identities that the local node can use to identify itself in RSCT HBA credentials, enter:

```
ctsvhbal
```

The output would be look like this:

```
ctsvhbal: The Host Based Authentication (HBA) mechanism identities for
the local system are:

                  Identity:  zathras.unixrulz.com

                  Identity:  9.127.100.101

ctsvhbal: At least one of the above identities must appear in the
trusted host list on the node where a service application resides in order
for client applications on the local system to authenticate successfully.
Make sure that at least one host name and one network address identity from the
above list appears in the trusted host list on the service systems used by
applications on this local system.
```

Related information

Commands: **ctskeygen**, **ctsthl**, **ctsvhbac**, **ctsvhbar**

Books: See the cluster security topics in *RSCT: Administration Guide*

ctsvhbar

Purpose

Returns the host name that the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism uses on the local node to verify credentials from a specified host.

Syntax

ctsvhbar [[**-d** | **-h** | **-m** | **-s**] | [**-e** *msgnum*[,*msgnum*...]] [**-l** { **1** | **2** | **3** | **4** } | **-b**] {*hostname* | *address*} [*hostname*... | *address*...]

Description

The **ctsvhbar** command is a verification utility for the RSCT host-based authentication (HBA) and enhanced host-based authentication (HBA2) security mechanisms. Use this command when you need to determine which host name the RSCT HBA or HBA2 security mechanism uses to verify credentials from a remote node.

The RSCT HBA or HBA2 security mechanism might use either a host name or a network address value as part of the identification information within a credential, depending on the method chosen by the application. If the local node is to service requests from remote nodes, at least one network address and host name for that remote node must appear in the trusted host list on the local node. To verify that the remote node can successfully authenticate the local node, system administrators can use a combination of RSCT cluster security commands:

1. On the local node and the remote node, issue the **ctsvhbac** command to verify that each node has a valid RSCT HBA or HBA2 security mechanism configuration.
2. On the local node, issue the **ctsvhbal** command to determine the values that the RSCT HBA or HBA2 security mechanism will use to identify this host to a remote node.
3. On the remote node, issue the **ctsvhbar** command, specifying the local node's host name or IP address, to determine the value that the remote node will use to verify RSCT HBA or HBA2 credentials transmitted from the local node.
4. Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two nodes are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these steps verifies successful authentication in one direction; in other words, the procedure verifies only that the remote node can authenticate requests from the local node. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local node can successfully authenticate the remote node. To complete the verification, the following additional steps are required:

- On the remote node, issue the **ctsvhbal** command to determine the values that the RSCT HBA or HBA2 security mechanism will use to identify that host to the local node.
- On the local node, issue the **ctsvhbar** command, specifying the remote node's host name or IP address, to determine the value that the local node will use to verify RSCT HBA or HBA2 credentials transmitted from the remote node.

- Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two nodes are using the same scheme for host-name resolution. If an exact host-name match does not appear in the output, repair the host-name resolution scheme, and repeat the steps above until both commands yield an exact match.

Completing these additional steps verifies successful authentication when traffic flows in the opposite direction, from the remote node to the local node.

For more detailed instructions and examples, see the cluster security topics in *RSCT: Administration Guide*.

Parameters

hostname The host name of a remote node.

address The network address of a remote node.

Flags

-b Produces brief output. When this option is used, the command displays the host identities provided by the command user, the fully qualified host identities obtained for them, and any errors. If the **-l** option is specified, this option is ignored.

-d Displays the list of probes required for successful execution of this command.

-e Specifies a list of error messages that are not to be displayed by this command during its execution. One or more message numbers may be specified. Message numbers must be in the xxxx-yyy format. Multiple messages are to be separated by commas (,) with no white space characters.

-h Displays a help message for this command.

-l Allows the Cluster System Management (CSM) Probe Infrastructure to set the detail level of the output. Accepted levels are:

- **1** Verbose mode. Displays the command purpose summary and status information for all tests.
- **2** Displays the command purpose summary and any attention or error conditions detected in any tests.
- **3** Displays any attention or error conditions detected in any tests.
- **4** Silent mode. Displays errors detected during the tests.

-m Displays a detailed description of the command and its purpose.

-s Displays a summary of the purpose for the command.

Exit status

Exit status conforms to the CSM Probe Infrastructure conventions.

0 No problems detected. Any messages displayed are informational. No administration intervention is required.

10 No problems were detected. The command was unable to resolve the host name or IP address provided by the command user. The command user should verify that the correct host name or IP address was used. If the

correct name or address was used, the system administrator should verify that the host-name resolution scheme used by the local node permits that name or address to be resolved.

127 Unexpected failure in this command.

Security

Permissions on the **ctsvhbar** command permit members of the **bin** user group to execute this command.

Implementation specifics

This command is part of the RSCT cluster security services. It is shipped as part of the **rsct.core.sec** fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsvhbar

Examples

To return the host name that the RSCT HBA or HBA2 security mechanism would use on the local node to verify credentials from the host identified by the host name **zathras**, you would enter:

```
ctsvhbar zathras
```

The output would look like this:

```
Host name or network address: zathras
Fully qualified host name
     used for authentication: zathras.unixrulz.com
```

To return the host name that the RSCT HBA or HBA2 security mechanism would use on the local node to verify credentials from the host identified by the network address **9.127.100.101**, you would enter:

```
ctsvhbar 9.127.100.101
```

The output would look like this:

```
Host name or network address: 9.127.100.101
Fully qualified host name
     used for authentication: epsilon3.unixrulz.com
```

To return the host name that the RSCT HBA or HBA2 security mechanism would use on the local node to verify credentials from both the host identified by the host name **zathras**, and the host identified by the network address **9.127.100.101**, you would enter:

```
ctsvhbar zathras 9.127.100.101
```

The output would look like this:

```
Host name or network address: zathras
Fully qualified host name
     used for authentication: zathras.unixrulz.com
Host name or network address: 9.127.100.101
Fully qualified host name
     used for authentication: epsilon3.unixrulz.com
```

Related information

Commands: **ctskeygen**, **ctsthl**, **ctsvhbac**, **ctsvhbal**

Books: See the cluster security topics in *RSCT: Administration Guide*

Part 7. Root command management

Chapter 13. Least-privilege (LP) resource manager commands 349
chlpcmd 350
lphistory 354
lslpcmd 359
mklpcmd 364
rmlpcmd 369
runlpcmd 372

Chapter 14. LP access control list (ACL) commands 377
chlpclacl 378
chlpracl 383
chlpriacl 389
chlprsacl 394
lslpclacl 399
lslpracl 404
lslpriacl 410
lslprsacl 415

Chapter 15. LP man pages 421
lpacl 422

© Copyright IBM Corp. 2002, 2007

Chapter 13. Least-privilege (LP) resource manager commands

© Copyright IBM Corp. 2002, 2007

chlpcmd

Purpose

Changes the attribute values of a least-privilege (LP) resource.

Syntax

To change the attribute values of an LP resource:

- On the local node:

  chlpcmd [**–l 0** | **1**] [**–c 0** | **1** | **2** | **3**] [**–h**] [**–TV**] *resource_name attr1*=*value1* [*attr2*=*value2*...]

  chlpcmd –r [**–h**] [**–TV**] *resource_name*

- On all nodes in a domain:

  chlpcmd –a [**–l 0** | **1**] [**–c 0** | **1** | **2** | **3**] [**–h**] [**–TV**] *resource_name attr1*=*value1* [*attr2*=*value2*...]

  chlpcmd –a –r [**–h**] [**–TV**] *resource_name*

- On a subset of nodes in a domain:

  chlpcmd –n *host1* [**,***host2***,**...] [**–l 0** | **1**] [**–c 0** | **1** | **2** | **3**] [**–h**] [**–TV**] *resource_name attr1*=*value1* [*attr2*=*value2*...]

  chlpcmd –n *host1* [**,***host2***,**...] **–r** [**–h**] [**–TV**] *resource_name*

Description

Use the **chlpcmd** command to change any of the read/write attribute values of an LP resource. 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). Use the **-r** flag to recalculate and assign the **CheckSum** attribute. Use the **-c** flag to change the **ControlFlags** attribute. Use the **-l** flag to change the **Lock** attribute. Use *attr*=*value* parameters to modify these attributes: **Name**, **CommandPath**, **RunCmdName**, **FilterScript**, **FilterArg**, and **Description**.

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

resource_name
: Specifies the name of the LP resource to change.

attr1=*value1* **[***attr2*=*value2***...]**
: Specifies one or more read/write attributes and their new values.

Flags

–a Changes attribute values for *resource_name* 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 **chlpcmd** 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, **chlpcmd –a** runs in the management domain. To run **chlpcmd –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–n *host1***[,***host2***,...]**
Specifies one or more nodes in the domain on which the LP resource is to be changed. By default, the LP resource is changed on the local node. This flag is valid only in a management domain or a peer domain. If the **CT_MANAGEMENT_SCOPE** environment 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 **chlpcmd** command runs once for the first valid scope that the LP resource manager finds.

–r Recalculates and assigns the **CheckSum** attribute value for this LP resource. Use the **-r** flag when:
- You have modified the command or script that this LP resource represents.
- You want to change the **CheckSum** value from **0** to the correct value after the command or script becomes available on the system.

–l 0 | 1
Locks or unlocks the resource. You can use this flag to protect the resource from being deleted by accident. The default value is **0**, which means no lock is set. To lock the resource, use **chlpcmd -l 1**.

–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.

For RSCT version 2.3.6.0 (AIX 5.2), RSCT version 2.4.2.0 (AIX 5.3), and later, the **ControlFlags** attribute value specifies whether the **CheckSum** value is to be validated.

In previous versions 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 versions 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.

–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.

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_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 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 **CT_MANAGEMENT_SCOPE** is not set, *local* scope is used.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output. 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.

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.

Security

To run the **chlpcmd** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- write 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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chlpcmd

Examples

1. To change the **Lock** attribute of LP resource **lpcommand1** before deleting a resource on a local node, enter:

   ```
   chlpcmd -l 0 lpcommand1
   ```

2. Suppose **nodeA** is in a management domain and **CT_MANAGEMENT_SCOPE** is set to **3**. To recalculate the **CheckSum** attribute value of LP resource **lpcommand2** on **nodeA**, enter:

   ```
   chlpcmd -r -n nodeA lpcommand2
   ```

Related information

- “lpacl ” on page 422, for general information about LP ACLs
- “rmccli ” on page 74, for general information about RMC commands and *attr*=*value* syntax

Books: *RSCT: Administration Guide* , for information about:
- modifying LP ACLs
- LP resource attributes and their definitions

Commands: **lphistory**, **lslpcmd**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

lphistory

Purpose

Displays or clears the history list of least-privilege (LP) commands that have been run.

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 of the 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 of the 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 the **-L** flag or the *num_records* parameter.

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.

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 will default 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 will default 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 flagis 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 (**/usr/sbin/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.

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_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 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.

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.

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.

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 “lpacl ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lphistory

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
   ```

Related information

“lpacl ” on page 422, for general information about LP ACLs

Books: *RSCT: Administration Guide*, for information about modifying LP ACLs

Commands: **chlpcmd**, **lslpcmd**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

lslpcmd

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.

Parameters

*resource_name1***[,***resource_name2***,...]**
Specifies one or more LP resources for which you want to display information.

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.

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_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 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.

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.

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.

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 “lpacl ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lslpcmd

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
------------------------------
⋮
```

Related information

“lpacl ” on page 422, for general information about LP ACLs

Books: *RSCT: Administration Guide*, for information about modifying LP ACLs

Commands: **chlpcmd**, **lphistory**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

mklpcmd

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.

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 “lpacl ” on page 422 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 "lpacl " on page 422 for descriptions of these permissions.

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.

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_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 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.

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.

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.

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 "lpacl " on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/mklpcmd

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
   ```

Related information

“lpacl ” on page 422, for general information about LP ACLs

Books: *RSCT: Administration Guide*, for information about modifying LP ACLs

Commands: **chlpcmd**, **lphistory**, **lslpcmd**, **mkrsrc**, **rmlpcmd**, **runlpcmd**

rmlpcmd

Purpose

Removes one or more least-privilege (LP) resources from the resource monitoring and control (RMC) subsystem.

Syntax

To remove one or more LP resources:

- From the local node:

  rmlpcmd [**–h**] [**–TV**] *resource_name1* [**,** *resource_name2* **,** ...]
- From all nodes in a domain:

  rmlpcmd –a [**–h**] [**–TV**] *resource_name1* [**,** *resource_name2* **,** ...]
- From a subset of nodes in a domain:

  rmlpcmd –n *host1* [**,***host2***,**...] [**–h**] [**–TV**] *resource_name1* [**,** *resource_name2* **,** ...]

Description

The **rmlpcmd** command removes one or more LP resources from the RMC 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). You can use the **rmlpcmd** command to remove LP resources from particular nodes or all nodes in a domain. If you want to remove locked LP resources, you must first use the **chlpcmd** command to unset the resource's **Lock** attribute.

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

*resource_name1***[,***resource_name2***,...]**
Specifies one or more LP resources to be removed.

Flags

–a Removes one or more LP resources from 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 **rmlpcmd** 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, **rmlpcmd –a** runs in the management domain. To run **rmlpcmd –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–n *host1***[,***host2***,...]**
Specifies one or more nodes in the domain from which the LP resource is to be removed. By default, the LP resource is removed from 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 **rmlpcmd** command runs once for the first valid scope that the LP resource manager finds.

–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.

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_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 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.

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.

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.

Security

To run the **rmlpcmd** command, you need read and write permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See “lpacl ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/rmlpcmd

Examples

1. To remove an LP resource named **LP1**, enter:
   ```
   rmlpcmd LP1
   ```
2. To remove LP resources **LP1** and **LP2**, enter:
   ```
   rmlpcmd LP1 LP2
   ```

Related information

“lpacl ” on page 422, for general information about LP ACLs

Books: *RSCT: Administration Guide*, for information modifying LP ACLs

Commands: **chlpcmd**, **lphistory**, **lslpcmd**, **mklpcmd**, **runlpcmd**

runlpcmd

Purpose

Runs a least-privilege (LP) resource.

Syntax

To run an LP resource:

- On the local node:

  runlpcmd –N *resource_name* | *RunCmdName* [**–h**] [**–TV**] [*"flags_and_parms"*]
- On all nodes in a domain:

  runlpcmd –a –N *resource_name* | *RunCmdName* [**–h**] [**–TV**] [*"flags_and_parms"*]
- On a subset of nodes in a domain:

  runlpcmd –n *host1* [**,***host2***,**...] **–N** *resource_name* | *RunCmdName* [**–h**] [**–TV**] [*"flags_and_parms"*]

Description

The **runlpcmd** command runs an LP resource, which is a **root** command or script to which users are granted access based on permissions in the LP access control lists (ACLs). You can use the **runlpcmd** command to call the LP command corresponding to a particular *RunCmdName* value with access permissions that match the permissions of the calling user. When **runlpcmd** is called with the **-N** flag, the LP command that is specified by the *resource_name* parameter is run. Specify all parameters and flag needed for command invocation using the *flags_and_parms* parameter. If this parameter is not specified, an empty string is passed to the LP command. This is the default.

If the **CheckSum** attribute value is **0**, **runlpcmd** returns an error if the **ControlFlags** value is set to check for **CheckSum**; otherwise, no errors are returned. If the **ControlFlag** attribute of the LP command was set to validate the **CheckSum** before the LP command was run, **runlpcmd** performs such a check. The command is run only if the calculated **CheckSum** matches the value of the corresponding **CheckSum** attribute. If the two do not match, the command is rejected. If, however, the **ControlFlags** attribute is set to the default value, **CheckSum** validation is not performed.

You can specify the *RunCmdName* parameter along with with the **-N** *resource_name* flag and parameter combination. However, one restriction applies when you use the *RunCmdName* parameter. If more than one resource matches the *RunCmdName* value and the permissions of the calling user, **runlpcmd** returns an error. If one match exists for the *RunCmdName* value and the the permissions of the calling user, **runlpcmd** *RunCmdName* returns successfully. In order to circumvent this restriction, **runlpcmd** also lets users run LP commands by specifying their unique names, using the **-N** *resource_name* flag and parameter combination.

Before calling the LP command, **runlpcmd** checks to see if a **FilterScript** value exists. If so, it passes the **FilterArg** value and the *flags_and_parms* parameter string specified on the command line to **FilterScript**. If **FilterScript** returns a **0**, **runlpcmd** calls the LP command. If **FilterScript** execution resulted in a non-zero

value, **runlpcmd** returns an error. If **FilterScript** was empty, **runlpcmd** performs some checks, as specified in **ControlFlags**, and then calls the LP command directly.

The output of this command may include *"***RC=***return_code"* as the last line.

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

RunCmdName
Specifies the name of the LP resource that you want to run on one or more nodes in the domain.

"flags_and_parms"
Specifies the flags and parameters that are required input for the LP command or script. If this parameter is not specified, an empty string is passed to the LP command. This is the default.

Flags

–a Changes one or more 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 **runlpcmd** 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, **runlpcmd –a** runs in the management domain. To run **runlpcmd –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 changed. 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 **runlpcmd** command runs once for the first valid scope that the LP resource manager finds.

–N *resource_name*
Specifies the name of the LP resource that you want to run on one or more nodes in the domain.

–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.

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 the environment variable 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_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 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.

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.

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.

Security

To run the **runlpcmd** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- execute 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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/runlpcmd

Examples

To run the LP resource called **LP1**, which has required input flags and parameters **-a -p User Group**, enter:

```
runlpcmd LP1 "-a -p User Group"
```

Related information

“lpacl ” on page 422, for general information about LP ACLs

Books: *RSCT: Administration Guide*, for information about modifying LP ACLs

Commands: **chlpcmd**, **lphistory**, **lslpcmd**, **mklpcmd**, **rmlpcmd**

Chapter 14. LP access control list (ACL) commands

© Copyright IBM Corp. 2002, 2007

chlpclacl

Purpose

Changes the access controls for the least-privilege (LP) resource class (**IBM.LPCommands**).

Syntax

To add one or more accesses to the **IBM.LPCommands** Class ACL or to overwrite the **IBM.LPCommands** Class ACL with one or more accesses:

chlpclacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o**] [**-h**] [**-TV**] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the **IBM.LPCommands** Class ACL or to overwrite the **IBM.LPCommands** Class ACL with one or more accesses all using the same permissions:

chlpclacl [**-a** | **-n** *host1*[**,***host2*,...]] **-l** [**-o**] [**-h**] [**-TV**] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the **IBM.LPCommands** Class ACL:

chlpclacl [**-a** | **-n** *host1*[**,***host2*,...]] **-d** [**-h**] [**-TV**] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the **IBM.LPCommands** Class ACL or to overwrite the **IBM.LPCommands** Class ACL, with the accesses specified in a file:

chlpclacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o** | **-d**] **-f** *file_name* [**-h**] [**-TV**]

To set the **IBM.LPCommands** Class ACL to deny all accesses:

chlpclacl [**-a** | **-n** *host1*[**,***host2*,...]] **-x** [**-h**] [**-TV**]

Description

The **chlpclacl** command changes the access control list (ACL) that is associated with the least-privilege (LP) resource class (**IBM.LPCommands**). This command allows an access to be added to or removed from the **IBM.LPCommands** Class ACL. This ACL controls access to such class operations as creating LP resources and deleting LP resources. One Class ACL exists on each node for the **IBM.LPCommands** class.

To add accesses to the **IBM.LPCommands** Class ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the **-l** flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the **-o** flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the Class ACL are deleted.

To delete accesses from the **IBM.LPCommands** Class ACL, use the **-d** flag and specify the IDs to be deleted.

Use the **-f** flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

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

ID Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see the **User identities** section of “lpacl ” on page 422.

perm Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

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 “lpacl ” on page 422 for descriptions of these permissions.

Flags

–a Changes **IBM.LPCommands** Class ACLs 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 **chlpclacl** 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, **chlpclacl –a** runs in the management domain. To run **chlpclacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–d Removes the ACL entry for the specified ID from the **IBM.LPCommands** Class ACL.

–f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

–l Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

–n *host1***[,***host2***,...]**
Specifies the nodes in the domain on which the **IBM.LPCommands** Class ACL should be changed. By default, the **IBM.LPCommands** Class ACL is changed 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 will run once for the first valid scope found.

–o Indicates that the specified accesses overwrite any existing ACL entries for the **IBM.LPCommands** Class ACL. Any ACL entries in the **IBM.LPCommands** Class ACL are deleted.

–x Sets the **IBM.LPCommands** Class ACL to deny all accesses to the **IBM.LPCommands** class attributes and class operations. Any ACL entries in the **IBM.LPCommands** Class ACL are deleted.

–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.

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 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'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.

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.

Security

To run the **chlpclacl** command, you need read and administrator permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl " on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chlpclacl

Examples

1. To give user **joe** on **nodeA** write permission to the **IBM.LPCommands** class so that he can create LP resources on **nodeA**, run one of these commands on **nodeA**:

   ```
   chlpclacl joe@NODEID w

   chlpclacl joe@LOCALHOST w
   ```

2. **nodeA** and **nodeB** are in a peer domain. To give user **joe** on **nodeB** write permission to the **IBM.LPCommands** class so that he can create LP resources on **nodeB**, run this command on **nodeA**:

```
chlpclacl -n nodeB joe@LOCALHOST  w
```

In this example, specifying **joe@NODEID** instead of **joe@LOCALHOST** gives **joe** on **nodeA** write permission to the **IBM.LPCommands** class on **nodeB**.

3. To give user **joe** on **nodeA** write permission to the **IBM.LPCommands** class and **bill** on **nodeA** administrator permission and write permission to the **IBM.LPCommands** class on **nodeA**, run this command on **nodeA**:

   ```
   chlpclacl joe@LOCALHOST w bill@LOCALHOST wa
   ```

4. To give user **joe** on **nodeA** administrator permission to the **IBM.LPCommands** class on **nodeA**, overwriting the current **IBM.LPCommands** Class ACL so that this is the only access allowed, run this command on **nodeA**:

   ```
   chlpclacl -o joe@LOCALHOST a
   ```

5. To give users **joe**, **bill**, and **jane** on **nodeA** read and write permissions to the **IBM.LPCommands** class on **nodeA**, run this command on **nodeA**:

   ```
   chlpclacl -l joe@LOCALHOST  bill@LOCALHOST  jane@LOCALHOST  rw
   ```

6. To delete access for **joe** on **nodeA** from the **IBM.LPCommands** class on **nodeA**, run this command on **nodeA**:

   ```
   chlpclacl -d  joe@LOCALHOST
   ```

7. To add a list of accesses that are in a file named **/mysecure/aclfile** on **nodeA** to the **IBM.LPCommands** class on **nodeA**, run this command on **nodeA**:

   ```
   chlpclacl -f /mysecure/aclfile
   ```

   The contents of **/mysecure/aclfile** on **nodeA** could be:

   ```
   joe@LOCALHOST     w
   bill@LOCALHOST    wa
   jane@LOCALHOST    rw
   ```

8. To deny all accesses to the **IBM.LPCommands** class on **nodeA**, run this command on **nodeA**:

   ```
   chlpclacl -x
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpracl**, **chlpriacl**, **chlprsacl**, **lslpclacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

chlpracl

Purpose

Changes the access controls for a least-privilege (LP) resource.

Syntax

To add one or more accesses to a Resource ACL or to overwrite a Resource ACL with one or more accesses:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o**] [**-r**] [**-h**] [**-TV**] *resource ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to a Resource ACL or to overwrite an Resource ACL with one or more accesses all using the same permissions:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] **-l** [**-o**] [**-r**] [**-h**] [**-TV**] *resource ID_1* [*ID_2*...] *perm*

To delete one or more accesses from a Resource ACL:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] **-d** [**-r**] [**-h**] [**-TV**] *resource ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) a Resource ACL or to overwrite a Resource ACL, with the accesses specified in a file:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o** | **-d**] **-f** *file_name* [**-r**] [**-h**] [**-TV**] *resource*

To set a Resource ACL so that no permissions are allowed, or to use the Resource Shared ACL:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] { **-b** | **-x** } [**-r**] [**-h**] [**-TV**] *resource*

To set all of the Resource ACLs so that no permissions are allowed, or to use the Resource Shared ACL:

chlpracl [**-a** | **-n** *host1*[**,***host2*,...]] { **-B** | **-X** } [**-h**] [**-TV**]

Description

The **chlpracl** command changes the access control list (ACL) that is associated with a least-privilege (LP) resource. This command allows an access to be added to or removed from the Resource ACL. This ACL controls access to such resource operations as listing attribute values and running LP commands. One Resource ACL exists for each LP resource.

For controlling access to the LP resource, three different types of Resource ACLs exist:

1. Resource ACL
2. Resource Initial ACL
3. Resource Shared ACL

The **chlpracl** command allows the Resource ACL to indicate that the Resource Shared ACL should be used in its stead to control access. For descriptions of these ACLs, see “lpacl ” on page 422.

To add an access to the Resource ACL, specify the name of the LP resource, the ID, and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the **-l** flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the **-o** flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource ACL, use the **-d** flag and specify the name of the LP resource and the IDs to be deleted.

Use the **-f** flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

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

resource
Specifies the name of the LP resource for which the Resource ACL is changed.

ID Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see “lpacl ” on page 422.

perm Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

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 “lpacl ” on page 422 for a description of each permission and how it applies.

Flags

–a Changes the Resource ACLs for *resource* 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 **chlpracl** 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, **chlpracl –a** runs in the management domain. To run **chlpracl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–b Bypasses the ACL for the specified LP resource. The Resource Shared ACL is used for access control for this LP resource. Any ACL entries in the Resource ACL are deleted.

–B Bypasses the ACLs for all LP resources. The Resource Shared ACL is used for access control for all LP resources. Any ACL entries in the Resource ACLs are deleted. One Resource Shared ACL exists for each **IBM.LPCommands** class (or node).

–d Removes the ACL entry for the specified ID from the specified Resource ACL.

–f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

–l Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

–n *host1***[,***host2***,...]**
Specifies the nodes in the domain on which the Resource ACL should be changed. By default, the Resource ACL is changed 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 will run once for the first valid scope found.

–o Indicates that the specified ACL accesses overwrite any existing ACL entries for the specified Resource ACL. Any ACL entries in the Resource ACL are deleted.

–r Indicates that *resource* is a ″typical″ RSCT resource handle. The resource handle must be enclosed in quotation marks. The Resource ACL of the resource handle is modified.

–x Sets the Resource ACL for the specified LP resource to deny all accesses to the LP resource. Any ACL entries in the Resource ACL are deleted.

–X Sets the Resource ACL of all LP resources to deny all accesses to the LP resource. Any ACL entries in the Resource ACLs are deleted.

–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.

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 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'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.

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.

Security

To run the **chlpracl** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- read and administrator permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if these permissions exist in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See “lpacl ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chlpracl

Examples

1. To give user **joe** on **nodeA** the ability to run the LP command **lpcommand1** on **nodeA**, run one of these commands on **nodeA**:

   ```
   chlpracl lpcommand1 joe@NODEID  x

   chlpracl lpcommand1 joe@LOCALHOST  x
   ```

2. **nodeA** and **nodeB** are in a peer domain. To give user **joe** on **nodeB** the ability to run the LP command **lpcommand1** on **nodeB**, run this command on **nodeA**:

   ```
   chlpracl -n nodeB lpcommand1 joe@LOCALHOST  x
   ```

   In this example, specifying **joe@NODEID** instead of **joe@LOCALHOST** gives **joe** on **nodeA** the ability to run the LP command **lpcommand1** on **nodeB**.

3. To give user **joe** on **nodeA** execute permission to the LP command **lpcommand1** and **bill** on **nodeA** administrator permission and write permission to the same resource on **nodeA**, run this command on **nodeA**:

   ```
   chlpracl lpcommand1 joe@LOCALHOST  x  bill@LOCALHOST  wa
   ```

4. To give user **joe** on **nodeA** administrator permission to the LP command **lpcommand1** on **nodeA**, overwriting the current ACLs for **lpcommand1** so that this is the only access allowed, run this command on **nodeA**:

   ```
   chlpracl -o lpcommand1 joe@LOCALHOST x
   ```

5. To give users **joe**, **bill**, and **jane** on **nodeA** the ability to run the LP command **lpcommand1** on **nodeA**, run this command on **nodeA**:

   ```
   chlpracl lpcommand1 -l joe@LOCALHOST  bill@LOCALHOST  jane@LOCALHOST  x
   ```

6. To delete access for **joe** on **nodeA** from the ACLs for the LP command **lpcommand1** on **nodeA**, run this command on **nodeA**:

   ```
   chlpracl -d lpcommand1 joe@LOCALHOST
   ```

7. To add a list of accesses that are in a file named **/mysecure/aclfile** on **nodeA** to the LP command **lpcommand1** on **nodeA**, run this command on **nodeA**:

```
chlpracl -f /mysecure/aclfile lpcommand1
```

The contents of **/mysecure/aclfile** on **nodeA** could be:

```
joe@LOCALHOST     x
bill@LOCALHOST                  ax
jane@LOCALHOST    wx
```

8. To bypass the Resource ACL for the LP command **lpcommand1** on **nodeA**, and use the Resource Shared ACL to control access to it, run this command on **nodeA**:

   ```
   chlpracl -b lpcommand1
   ```

9. To bypass the Resource ACLs for all of the LP resources on **nodeA**, and use the Resource Shared ACL to control accesses, run this command on **nodeA**:

   ```
   chlpracl -B
   ```

10. To deny all accesses to the LP command **lpcommand1** on **nodeA**, run this command on **nodeA**:

    ```
    chlpracl -x lpcommand1
    ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpriacl**, **chlprsacl**, **lslpclacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

chlpriacl

Purpose

Changes the access controls for the least-privilege (LP) Resource Initial ACL.

Syntax

To add one or more accesses to the Resource Initial ACL or to overwrite the Resource Initial ACL with one or more accesses:

chlpriacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o**] [**-h**] [**-TV**] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the Resource Initial ACL or to overwrite the Resource Initial ACL with one or more accesses all using the same permissions:

chlpriacl [**-a** | **-n** *host1*[**,***host2*,...]] **-l** [**-o**] [**-h**] [**-TV**] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the Resource Initial ACL:

chlpriacl [**-a** | **-n** *host1*[**,***host2*,...]] **-d** [**-h**] [**-TV**] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the Resource Initial ACL or to overwrite the Resource Initial ACL, with the accesses specified in a file:

chlpriacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o** | **-d**] **-f** *file_name* [**-h**] [**-TV**]

To set the Resource Initial ACL to use the Resource Shared ACL or so that no permissions are allowed:

chlpriacl [**-a** | **-n** *host1*[**,***host2*,...]] { **-b** | **-x** } [**-h**] [**-TV**]

Description

The **chlpriacl** command changes the access control list (ACL) that is associated with the least-privilege (LP) Resource Initial ACL. This command allows a user to be added to or removed from the Resource Initial ACL. This ACL is used to initialize a Resource ACL when the LP resource is created. The Resource Initial ACL can consist of ACL entries that define permissions to the LP resource or it can indicate that the Resource Shared ACL should be used to control access instead of the Resource ACL. One Resource Initial ACL exists on each node for the **IBM.LPCommands** class.

To add accesses to the Resource Initial ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the **-l** flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the **-o** flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource Initial ACL, use the **-d** flag and specify the IDs to be deleted.

Use the **-f** flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

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

ID Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see “lpacl ” on page 422.

perm Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

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 “lpacl ” on page 422 for a description of each permission and how it applies.

Flags

–a Changes the Resource Initial ACLs 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 **chlpriacl** 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, **chlpriacl –a** runs in the management domain. To run **chlpriacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–b Sets the Resource Initial ACL to indicate that the Resource ACL is bypassed and that the Resource Shared ACL is used for access control for

the LP resource. Any ACL entries in the Resource Initial ACL are deleted. When a new LP resource is created, the Resource Shared ACL is used for it.

–d Removes the ACL entry for the specified ID from the Resource Initial ACL.

–f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

–l Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

–n *host1***[,***host2***,...]**
Specifies the node in the domain on which the Resource Initial ACL should be changed. By default, the Resource Initial ACL is changed 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 will run once for the first valid scope found.

–o Indicates that the specified ACL entries overwrite any existing ACL entries for the Resource Initial ACL. Any ACL entries in the Resource Initial ACL are deleted.

–x Sets the Resource Initial ACL to deny all accesses to the LP resource. Any ACL entries in the Resource Initial ACL are deleted. When a new LP resource is created, all accesses will be denied to it.

–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.

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 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'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.

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.

Security

To run the **chlpriacl** command, you need read and administrator permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl " on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chlpriacl

Examples

1. To give user **joe** on **nodeA** execute permission in the Resource Initial ACL on **nodeA**, run one of these commands on **nodeA**:

```
chlpriacl joe@NODEID  x

chlpriacl joe@LOCALHOST  x
```

2. **nodeA** and **nodeB** are in a peer domain. To give user **joe** on **nodeB** execute permission to the Resource Initial ACL on **nodeB**, run this command on **nodeA**:

   ```
   chlpriacl -n nodeB joe@LOCALHOST  x
   ```

   In this example, specifying **joe@NODEID** instead of **joe@LOCALHOST** gives **joe** on **nodeA** execute permission to the Resource Initial ACL on **nodeB**.
3. To give user **joe** on **nodeA** execute permission and **bill** on **nodeA** administrator permission and read permission to the Resource Initial ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl joe@LOCALHOST  x  bill@LOCALHOST  ra
   ```
4. To give user **joe** on **nodeA** execute permission to the Resource Initial ACL on **nodeA**, overwriting the current ACLs so that this is the only access allowed, run this command on **nodeA**:

   ```
   chlpriacl -o joe@LOCALHOST x
   ```
5. To give users **joe**, **bill**, and **jane** on **nodeA** read permission and write permission to the Resource Initial ACL on **nodeA** on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl -l joe@LOCALHOST  bill@LOCALHOST jane@LOCALHOST  rw
   ```
6. To delete access for **joe** on **nodeA** from the Resource Initial ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl -d  joe@LOCALHOST
   ```
7. To add a list of accesses that are in a file named **/mysecure/aclfile** on **nodeA** to the Resource Initial ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl -f /mysecure/aclfile
   ```

   The contents of **/mysecure/aclfile** on **nodeA** could be:

   ```
   joe@LOCALHOST     x
   bill@LOCALHOST    rw
   jane@LOCALHOST    rwa
   ```
8. To set the Resource Initial ACL on **nodeA** so it indicates that the Resource Shared ACL on **nodeA** is used to control accesses for newly-created LP resources on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl -b
   ```
9. To set the Resource Initial ACL on **nodeA** so that it denies all accesses for newly-created LP resources on **nodeA**, run this command on **nodeA**:

   ```
   chlpriacl -x
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlprsacl**, **lslpclacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

chlprsacl

Purpose

Changes the access controls for the least-privilege (LP) Resource Shared ACL.

Syntax

To add one or more accesses to the Resource Shared ACL or to overwrite the Resource Shared ACL with one or more accesses:

chlprsacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o**] [**-h**] [**-TV**] *ID_1 perm1* [*ID_2 perm2*] ...

To add one or more accesses to the Resource Shared ACL or to overwrite the Resource Shared ACL with one or more accesses all using the same permissions:

chlprsacl [**-a** | **-n** *host1*[**,***host2*,...]] **-l** [**-o**] [**-h**] [**-TV**] *ID_1* [*ID_2*...] *perm*

To delete one or more accesses from the Resource Shared ACL:

chlprsacl [**-a** | **-n** *host1*[**,***host2*,...]] **-d** [**-h**] [**-TV**] *ID_1* [*ID_2*...]

To add accesses to (or remove accesses from) the Resource Shared ACL or to overwrite the Resource Shared ACL, with the accesses specified in a file:

chlprsacl [**-a** | **-n** *host1*[**,***host2*,...]] [**-o** | **-d**] **-f** *file_name* [**-h**] [**-TV**]

To set the Resource Shared ACL so that no permissions are allowed:

chlprsacl [**-a** | **-n** *host1*[**,***host2*,...]] **-x** [**-h**] [**-TV**]

Description

The **chlprsacl** command changes the access control list (ACL) that is associated with the Resource Shared ACL. This command allows a user to be added to or removed from the Resource Shared ACL. This ACL:

- is used to control accesses to LP resources when the Resource ACL indicates that it (the Resource Shared ACL) has control
- can control access to one or more LP resources
- can consist of ACL entries that define permissions to the LP resources

One Resource Shared ACL exists on each node for the **IBM.LPCommands** class.

The **chlpracl** command is used to indicate that the access to an LP resource is controlled by the Resource Shared ACL. The **chlpriacl** command is used to indicate that accesses to newly-created LP resources are controlled by the Resource Shared ACL, by modifying the Resource Initial ACL.

To add accesses to the Resource Shared ACL, specify the ID and the permission the ID is to have. More than one ID and permission pair can be specified. If you want to add multiple IDs and they will all have the same permission, use the **-l** flag to indicate that the format of the command is a list of IDs followed by a single permission that applies to all of the IDs. If you use the **-o** flag, the IDs and permissions specified with the command will overwrite the existing accesses. The previously-defined accesses in the ACL are deleted.

To delete accesses from the Resource Shared ACL, use the **-d** flag and specify the IDs to be deleted.

Use the **-f** flag to indicate that the accesses are specified in a file. Each line of the file will be an ID and permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

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

ID Specifies the network identity of the user. If the same *ID* is listed more than once, the last permission specified is used. For a description of how to specify the network identity, see “lpacl ” on page 422.

perm Specifies the permission allowed for *ID*. *perm* is specified as a string of one or more characters, where each character represents a particular permission. The valid values for *perm* are:

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 “lpacl ” on page 422 for a description of each permission and how it applies.

Flags

–a Changes the Resource Shared ACLs 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 **chlprsacl** 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, **chlprsacl –a** runs in the management domain. To run **chlprsacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

–d Removes the ACL entry for the specified ID from the Resource Shared ACL.

–f *file_name*
Indicates that the accesses are specified in *file_name*. Each line of this file consists of an ID and the permission for that ID. If the **-d** flag is used with the **-f** flag, only the ID is needed on each line. Everything after the first space is ignored.

–l Indicates that there is a list of IDs followed by a single permission that is used for all of the IDs.

–n *host1***[,***host2***,...]**
Specifies the node in the domain on which the Resource Shared ACL should be changed. By default, the Resource Shared ACL is changed 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 will run once for the first valid scope found.

–o Indicates that the specified ACL entries overwrite any existing ACL entries for the Resource Shared ACL. Any ACL entries in the Resource Shared ACL are deleted.

–x Sets the Resource Shared ACL to deny all accesses to the LP resources that use the Resource Shared ACL. Any ACL entries in the Resource Shared ACL are deleted.

–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.

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 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'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.

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.

Security

To run the **chlprsacl** command, you need read and administrator permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl " on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/chlprsacl

Examples

1. To give user **joe** on **nodeA** execute permission in the Resource Shared ACL on **nodeA**, run one of these commands on **nodeA**:

```
chlprsacl joe@NODEID  x

chlprsacl joe@LOCALHOST  x
```

2. **nodeA** and **nodeB** are in a peer domain. To give user **joe** on **nodeB** execute permission to the Resource Shared ACL on **nodeB**, run this command on **nodeA**:

   ```
   chlprsacl -n nodeB joe@LOCALHOST  x
   ```

   In this example, specifying **joe@NODEID** instead of **joe@LOCALHOST** gives **joe** on **nodeA** execute permission to the Resource Shared ACL on **nodeB**.
3. To give user **joe** on **nodeA** execute permission and **bill** on **nodeA** administrator permission and read permission to the Resource Shared ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlprsacl joe@LOCALHOST  x  bill@LOCALHOST  ra
   ```
4. To give user **joe** on **nodeA** execute permission to the Resource Shared ACL on **nodeA**, overwriting the current ACLs so that this is the only access allowed, run this command on **nodeA**:

   ```
   chlprsacl -o joe@LOCALHOST x
   ```
5. To give users **joe**, **bill**, and **jane** on **nodeA** read permission and write permission to the Resource Shared ACL on **nodeA** on **nodeA**, run this command on **nodeA**:

   ```
   chlprsacl -l joe@LOCALHOST  bill@LOCALHOST jane@LOCALHOST  rw
   ```
6. To delete access for **joe** on **nodeA** from the Resource Shared ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlprsacl -d  joe@LOCALHOST
   ```
7. To add a list of accesses that are in a file named **/mysecure/aclfile** on **nodeA** to the Resource Shared ACL on **nodeA**, run this command on **nodeA**:

   ```
   chlprsacl -f /mysecure/aclfile
   ```

   The contents of **/mysecure/aclfile** on **nodeA** could be:

   ```
   joe@LOCALHOST     x
   bill@LOCALHOST    rw
   jane@LOCALHOST    rwa
   ```
8. To set the Resource Shared ACL on **nodeA** so that it denies all accesses for LP resources that use it on **nodeA**, run this command on **nodeA**:

   ```
   chlprsacl -x
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **lslpclacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

lslpclacl

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 “lpacl ” on page 422 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 See “lpacl ” on page 422 for a description of each 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'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 **lslpclacl** 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, **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 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 **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 will run 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'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.

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 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'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.

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.

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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lslpclacl

Examples

1. To list the **IBM.LPCommands** Class ACLs on **nodeA** in table format, run this command on **nodeA**:

   ```
   lslpclacl
   ```

   The output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

```
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 output will look like this:

```
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 information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **chlprsacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

lslpracl

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 ” on page 422 for a description of the network identity.
Identity	The network identity of the user. See “lpacl ” on page 422 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

See “lpacl ” on page 422 for a description of each 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’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 **lslpracl** 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, **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 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 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 will run 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'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.

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 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'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.

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.

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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/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 output will look like this:

   ```
   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 output will look like this:

```
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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   Resource ACLs for LPRM
   Name            Identity                Permissions     NodeName
   lpcommand2      bill@0x374bdcbe384ed38a rx              nodeA
   lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **chlprsacl**, **lslpcmd**, **lslpclacl**, **lslpriacl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

lslpriacl

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 ” on page 422 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 See “lpacl ” on page 422 for a description of each 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'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 **lslpriacl** 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, **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 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 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 will run 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'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.

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 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'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.

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.

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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lslpriacl

Examples

1. To list the Resource Initial ACLs on **nodeA** in table format, run this command on **nodeA**:

   ```
   lslpriacl
   ```

   The output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

```
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 output will look like this:

```
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 information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **chlprsacl**, **lslpcmd**, **lslpclacl**, **lslpracl**, **lslprsacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

lslprsacl

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 ” on page 422 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 See “lpacl ” on page 422 for a description of each 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'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 **lslprsacl** 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, **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 will run 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'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.

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 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'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.

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.

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 ” on page 422 for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/lslprsacl

Examples

1. To list the Resource Shared ACLs on **nodeA** in table format, run this command on **nodeA**:

   ```
   lslprsacl
   ```

   The output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

   ```
   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 output will look like this:

```
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 output will look like this:

```
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 information

Books: *RSCT: Administration Guide*, for information about:

- the least-privilege (LP) resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **chlprsacl**, **lslpcmd**, **lslpclacl**, **lslpracl**, **lslpriacl**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

“lpacl ” on page 422, for general information about LP ACLs

Chapter 15. LP man pages

© Copyright IBM Corp. 2002, 2007

lpacl

Purpose

Provides general information about protecting the least-privilege (LP) commands resource class and its resources 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.

   For more information about the **ctrmc.acls** file and the ACLs it defines, see “ctrmc.acls ” on page 165 and the *RSCT: Administration Guide*.
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 man page 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/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 Chapter 13, “Least-privilege (LP) resource manager commands,” on page 349. For information about how to use these commands, see the *RSCT: Administration Guide*.

Because each LP resource can define a unique command, RMC implements ACLs for the **IBM.LPCommands** class that allow 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

For descriptions of these commands, see Chapter 14, “LP access control list (ACL) commands,” on page 377 and “mklpcmd ” on page 364. For information about how to use these commands, see the *RSCT: Administration Guide*.

Basic ACL structure

Typically, an ACL is composed of a list of ACL entries. Each ACL entry specifies an identity and a set of permissions granted to that identity. The complete list of ACL entries determines how the ACL controls access to the associated class or resource.

A resource-associated ACL can refer to another ACL instead of containing a list of ACL entries itself. When a resource-associated ACL refers to another ACL, the set of ACL entries in the referenced ACL controls access to the resource.

Types of ACLs

Four types of ACLs control access to the **IBM.LPCommands** class and its resources, as follows:

Class ACL

A *Class ACL* controls access to class operations on one node. You need to have been granted specific permissions to perform class operations, such as listing class attributes, creating class resources, and deleting class resources.

A Class ACL is composed of a list of ACL entries. The list of ACL entries controls access to class operations on the node. If the list is empty, no identity is permitted to perform class operations on the node.

When you try to perform a class operation on the **IBM.LPCommands** class on a node — creating a new resource, for example — RMC checks the Class ACL on that node to verify that you have the required permission to perform the operation. If you do not have the required permission, the operation is rejected.

One Class ACL exists on each node for the **IBM.LPCommands** class. Each node's Class ACL controls access to all **IBM.LPCommands** class operations on that node.

Resource ACL

A *Resource ACL* controls access to resource operations for one LP resource. You need to have been granted specific permissions to perform resource operations, such as listing resource attributes, modifying resource attributes, and running resource commands.

A Resource ACL can be composed of a list of ACL entries. In this case, the list of ACL entries controls access to resource operations for that resource. If the list is empty, no identity is permitted to perform resource operations for the resource.

A Resource ACL can refer to the Resource Shared ACL instead of containing a list of ACL entries itself. In this case, the list of ACL entries in the Resource Shared ACL controls access to resource operations for the resource. If the list is empty, no identity is permitted to perform resource operations for the resource.

When you try to perform a resource operation on an LP resource — running an LP command, for example — RMC first checks the Resource ACL for the selected resource to determine whether the Resource ACL contains a list of ACL entries or whether it refers to the Resource Shared ACL. If the Resource ACL has a list of ACL entries, RMC checks that list to verify that you have the required permission to perform the operation. If you do not have the required permission, the operation is rejected.

If the Resource ACL refers to the Resource Shared ACL, RMC checks the Resource Shared ACL to verify that you have the required permission to perform the operation. If you do not have the required permission, the operation is rejected.

One Resource ACL exists for each LP resource. When a Resource ACL refers to the Resource Shared ACL, the Resource Shared ACL that is being referenced is the one on the same node as the resource.

Resource Initial ACL

A *Resource Initial ACL* defines the initial contents of a Resource ACL created on a node.

Because a Resource Initial ACL is used to initialize Resource ACLs, a Resource Initial ACL can contain a list of ACL entries or a reference to the Resource Shared ACL.

When a new LP resource is created, its Resource ACL is initialized as specified by the Resource Initial ACL on the node.

One Resource Initial ACL exists on each node for the **IBM.LPCommands** class.

Resource Shared ACL

A *Resource Shared ACL* can control access to resource operations for multiple resources on one node.

A Resource Shared ACL is composed of a list of ACL entries. The list of ACL entries controls access to resource operations for all of the resources on the node that refer to the Resource Shared ACL. As with the other ACL types, the list of ACL entries can be empty.

To use this ACL, place ACL entries in it as you would in a Resource ACL. Then, modify the Resource ACLs on the same node to refer to the Resource Shared ACL. Using the Resource Shared ACL, you can use one list of ACL entries to control access to multiple resources on the same node.

One Resource Shared ACL exists on each node for the **IBM.LPCommands** class.

ACL entries

An RMC ACL for LP commands specifies a list of ACL entries. Each ACL entry defines a user identity and that identity's user permissions. A *user identity* is an authenticated network identity. The *user permissions* specify the access that a user has to the class or to the resources.

User identities: In an RMC ACL entry, the user identity can be in one of the following three forms:

1. [**host:**]*host_user_identifier*

   Specifies a host user identifier. The optional **host:** keyword specifies that the user identifier can be matched against a network identifier that is provided by the RSCT host-based authentication (HBA) or enhanced host-based authentication (HBA2) security mechanism. If the **host:** keyword is omitted and the entry does not take one of the other forms described, the entry is assumed to be a host user identifier. The host user identifier can be in one of the following three forms:

   a. *user_name@host_identifier*

      Specifies a particular authenticated user. You can specify *host_identifier* in several different formats. These formats, which are the same as when the host user identifier format is specified as a host identifier alone, are described as follows.

   b. *host_identifier*

      Specifies any authenticated user on the host identified. The host identifier can be:

      - a fully-qualified host name.
      - a short host name.
      - an IP address.
      - the RSCT node ID. This is the 16-digit hexadecimal number, for example: **0xaf58d41372c47686**.
      - the keyword **LOCALHOST**. This keyword is a convenient shorthand notation for the RSCT node ID of the node where the ACL exists. The **LOCALHOST** keyword is stored in the ACL.
      - the keyword **NODEID**. This keyword is a convenient shorthand notation for the RSCT node ID of the node where an ACL editing command is running. The **NODEID** keyword is not stored in the ACL; the node ID that the keyword represents is actually stored in the ACL..

   c. *"*****"*

      Specifies any authenticated user on any host. The asterisk (*****) must be enclosed in double quotation marks when it is specified as command input.

2. **none:***mapped_user_identifier*

   Specifies a mapped name, as defined in the **ctsec_map.global** file or the **ctsec_map.local** file. For more information, see "ctsec_map.global, ctsec_map.local" on page 167. For information about mapped user identifiers, see the *RSCT: Administration Guide*.

3. **UNAUTHENT**
   Specifies any unauthenticated user.

Some typical forms of a user identity are:

user@full_host_name
user@short_host_name
user@ip_address
user@node_ID
user@**LOCALHOST**
full_host_name
short_host_name
IP_address
node_ID
LOCALHOST
*

When you are running LP commands, the *host_identifier* parameter is often expected to be the RSCT node ID. You can use the **NODEID** keyword to represent the node ID of the node on which the command runs. To determine the node ID otherwise, enter:

```
lsrsrc IBM.Host NodeIDs
```

To determine all of the node IDs in a management domain or a peer domain, enter:

```
lsrsrc -ta IBM.Host NodeIDs NodeNameList
```

The node ID is displayed in decimal format. Use the hexadecimal equivalent for the *host_identifier*, preceded by **0x**. If the nodes are in a peer domain, enter:

```
lsrpnode -i
```

The node ID is displayed in hexadecimal. To use this value in the commands, you need to precede this value with **0x**. If the **CT_CONTACT** environment variable is used to specify where the RMC session occurs, the host_identifier is expected to be a fully-qualified host name, a short host name, or an IP address.

User permissions: The user permissions are expressed as a string of one or more characters, each representing a particular permission.

To compensate for the fine granularity of the permission set, RSCT provides two composite permissions. The **r** permission consists of individual permissions that allow ″read″ types of operations. The **w** permission consists of individual permissions that allow ″write″ types of operations. Most ACL entries will probably use these convenient composite permissions.

The permission set: The next two sections show two different views of the defined permission set. The first section describes the permission set using the composite permissions. The second section describes the permission set using the individual permissions.

Using composite permissions

r Read permission.
- To view the resource attribute values for an LP resource, you need this permission for the LP resource.
- To view the **IBM.LPCommands** class attribute values, you need this permission for the **IBM.LPCommands** class.

- You need this permission to list the LP ACLs.

Therefore, this permission is meaningful for any LP ACL. Read permission consists of the **q**, **l**, **e**, and **v** permissions.

w Write permission.

- To change the resource attribute values for an LP resource, you need this permission for the LP resource.
- To change the class attribute values for the **IBM.LPCommands** class, you need this permission for the **IBM.LPCommands** class.
- To create or delete LP resources, you need this permission for the **IBM.LPCommands** class.

Therefore, this permission is meaningful for any LP ACL. Write permission consists of the **d**, **s**, **c**, and **o** permissions.

a Administrator permission.

- To change the Resource ACL for an LP resource, you need this permission for the LP resource.
- To change the Class ACL, the Resource Initial ACL, or the Resource Shared ACL, you need this permission for the **IBM.LPCommands** class.

Therefore, this permission is meaningful for any LP ACL.

x Execute permission. To run an LP command that is defined in an LP resource, you need this permission for the LP resource. Therefore, this permission is meaningful for the LP Resource ACL, the Resource Initial ACL, and the Resource Shared ACL.

0 No permission. This permission denies you access to an LP resource or to the **IBM.LPCommands** class. Therefore, this permission is meaningful for any LP ACL.

Using individual permissions

q Query permission.

- To query the resource attribute values for an LP resource, you need this permission for the LP resource.
- To query the class attribute values, you need this permission for the **IBM.LPCommands** class.
- You need this permission to list the LP ACLs.

Therefore, this permission is meaningful for any LP ACL.

l Enumerate permission. To list the LP resources, you need this permission for the **IBM.LPCommands** class. Therefore, this permission is meaningful for the Class ACL.

e Event permission. To register, unregister, or query events, you need this permission for an LP resource or for the **IBM.LPCommands** class. Therefore, this permission is meaningful for any LP ACL.

v Validate permission. You need this permission to validate that an LP resource handle still exists. Therefore, this permission is meaningful for the Resource ACL, the Resource Initial ACL, and the Resource Shared ACL.

d Define and undefine permission. To create or delete LP resources, you need this permission for the **IBM.LPCommands** class. Therefore, this permission is meaningful for the Class ACL.

c Refresh permission. To refresh the **IBM.LPCommands** class configuration,

you need this permission for the **IBM.LPCommands** class. Therefore, this permission is meaningful for the Class ACL.

s Set permission.

- To set resource attribute values for an LP resource, you need this permission for the LP resource.
- To set class attribute values, you need this permission for the **IBM.LPCommands** class.

Therefore, this permission is meaningful for any LP ACL.

o Online, offline, and reset permission. Because LP resources do not support the online, offline, and reset operations, this permission has no meaning in LP ACLs.

a Administrator permission.

- To change the Resource ACL for an LP resource, you need this permission for the LP resource.
- To change the Class ACL, the Resource Initial ACL, or the Resource Shared ACL, you need this permission for the **IBM.LPCommands** class.

Therefore, this permission is meaningful for any LP ACL.

x Execute permission. To run an LP command that is defined in an LP resource, you need this permission for the LP resource. Therefore, this permission is meaningful for the LP Resource ACL, the Resource Initial ACL, and the Resource Shared ACL.

0 No permission. This permission denies you access to an LP resource or to the **IBM.LPCommands** class. Therefore, this permission is meaningful for any LP ACL.

Some permission characters have no meaning in certain types of ACLs. For example, the **l** permission has no meaning in a Resource ACL. A permission character that has no meaning in a certain type of ACL can be present in the ACL with no ill effect. For example, the **l** permission can be specified in an ACL entry of a Resource ACL. The presence of meaningless permissions in ACL entries is inevitable when the composite permissions are used.

In addition to the permissions that are granted explicitly by ACL entries, the **root** mapped identity always has query and administrator permission for ACL operations. If an ACL is set so that all access is denied, the **root** mapped identity can still be used to change the ACL, due to its implicit authority.

The system administrator needs to determine how ACLs should be defined for the **IBM.LPCommands** class and its resources. This depends on which operations users are required to perform.

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 man page is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/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 forth. In a management domain, it is most likely that the LP resources will be defined on the management server and the LP commands themselves will be targeted to the managed nodes. In these examples, the Resource Shared ACL is not used because separate permissions are desired for the individual LP resources. These examples assume that the LP resources have not yet been defined using the **mklpcmd** command.

1. You want to define the **lpadmin** ID to be the administrator for the LP commands. This ID will have 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 set this up, 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 will have administrator, read, and write permission to it.
2. The **lpadmin** ID can now create LP resources that define the LP commands that are needed. See “mklpcmd ” on page 364 for a description on how to create the LP resources. 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** will be 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 gives **joe** on the management server execute permission 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 to let other users 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 execute permission. In addition, **joe** needs to have execute 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 output will look like this:

   ```
   Name    OpState RSCTVersion NodeNum NodeID
   nodeA   Online  2.4.2.0     2       48ce221932ae0062
   nodeB   Online  2.4.2.0     1       7283cb8de374d123
   nodeC   Online  2.4.2.0     4       b3eda8374bc839de
   nodeD   Online  2.4.2.0     5       374bdcbe384ed38a
   nodeE   Online  2.4.2.0     2       ba74503cea374110
   nodeF   Online  2.4.2.0     1       4859dfbd44023e13
   nodeG   Online  2.4.2.0     4       68463748bcc7e773
   ```

   Then, to give **joe** the permissions as stated above, run on nodeB:

   ```
   chlpracl SysCmd1 -l joe@LOCALHOST joe@0x48ce221932ae0062 \
   joe@0x374bdcbe384ed38a x
   ```

Related information

Books: *RSCT: Administration Guide*, for information about:
- the LP resource manager
- how to use ACLs

Commands: **chlpclacl**, **chlpracl**, **chlpriacl**, **chlprsacl**, **chlpcmd**, **chrmcacl**, **lphistory**, **lslpclacl**, **lslpcmd**, **lslpracl**, **lslpriacl**, **lslprsacl**, **lsrpnode**, **lsrsrc**, **mklpcmd**, **rmlpcmd**, **runlpcmd**

Files: **ctrmc.acls**, **ctsec_map.global**, **ctsec_map.local**

Part 8. Topology services and group services

Chapter 16. Commands for subsystem control and status 433
cthactrl 434
nlssrc 436

Chapter 17. Topology services commands 439
cthatsctrl 440
cthatstune 443
hatsoptions 446
topsvcs 448
topsvcsctrl 449

Chapter 18. Group services commands 453
cthagsctrl 454
cthagstune 458
grpsvcsctrl 460
hagsd 464
hagsns 465
hagsvote 467

© Copyright IBM Corp. 2002, 2007

Chapter 16. Commands for subsystem control and status

© Copyright IBM Corp. 2002, 2007

cthactrl

Purpose

Controls subsystems within a cluster.

Syntax

cthactrl -i *init_opt* | **-s** | **-k** | **-b** | **-r** | **-d** | **-z** | **-h**

Description

The **cthactrl** command establishes and controls cluster subsystem information and manages topology services and group services.

Flags

-i *init_opt*
: Initializes the group services and topology services subsystems, where *init_opt* can be specified as:

  -c *cluster_name*
  : Specifies the cluster name.

  -n *nodenum*
  : Specifies the node number.

  -e *environ*
  : Specifies the subdirectory that contains the cluster access modules.

  -p *portspec*
  : Specifies the UDP port numbers for group services and topology services.

-s
: Starts the group services and topology services subsystems.

-k
: Stops the group services and topology services subsystems.

-b
: Rebuilds the group services and topology services subsystems' configurations (**machines.lst**, for example).

-r
: Refreshes the group services and topology services subsystems.

-d
: Deletes the group services and topology services subsystems.

-z
: Deinstalls the group services and topology services subsystems.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Exit status

0
: Successful completion.

a non-zero value
: A failure has occurred.

Security

You must have **root** authority to run this command.

Restrictions

This command applies to the **cthags** and **cthats** subsystems only.

Location

/usr/sbin/rsct/bin/cthactrl Contains the **cthactrl** command

Examples

1. To initialize the local node as a part of the cluster of **filesys1** and designate **12347** as the UDP port number for **cthags** and **12348** as the UDP port number for **cthags**, enter:

   ```
   cthactrl -i -c filesys1 -n 1 -p "cthats=12347,cthags=12348" -e filesys1
   ```

2. To start the group services and topology services subsystems (**cthags** and **cthats**), enter:

   ```
   cthactrl -s
   ```

3. To stop the group services and topology services subsystems (**cthags** and **cthats**), enter:

   ```
   cthactrl -k
   ```

Related information

Commands: **cthagsctrl**, **cthagstune**, **cthatsctrl**, **cthatstune**, **lssrc**, **nlssrc**

nlssrc

Purpose

Displays the status of a subsystem or a group of subsystems in canonical form.

Syntax

To display all status:

nlssrc [**–h** *host*] **–a**

To display group status:

nlssrc [**–h** *host*] **–g** *group_name*

To display subsystem status in canonical form:

nlssrc [**–h** *host*] [**–l**] [**–c**] **–s** *subsystem_name*

To display status by PID in canonical form:

nlssrc [**–h** *host*] [**–l**] [**–c**] **–p** *subsystem_pid*

Description

Use the **nlssrc** command to display the status of a subsystem or a group of subsystems in canonical form. Use **nlssrc -c** to get language-independent output for supported subsystems from the **lssrc** command. The status is displayed in English regardless of the installed language locale. If the **–c** flag is not specified, the **nlssrc** command will call the **lssrc** command that uses the daemon's locale.

Flags

–a Displays the current status of all defined subsystems.

–c Requests the canonical **lssrc** output of the supported subsystems.

–g *group_name*
Specifies a group of subsystems for which to get status. The command will fail if the subsystem object class does not contain *group-name*.

–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 (SRC) requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **–r** flag and the **/etc/hosts.equiv** file or the **.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.

–p *subsystem_pid*
Specifies a particular instance of the *subsystem_pid* parameter to get status for, or a particular instance of the subsystem to which the status subserver request is to be taken.

–s *subsystem_name*
Specifies a subsystem for which to get status. The *subsystem_name* parameter can be the actual subsystem name or the synonym name for the subsystem. The command will fail if the subsystem object class does not contain *subsystem*.

Security

You do *not* need **root** authority to run this command.

Restrictions

This command applies to the **cthags** and **cthats** subsystems only.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/nlssrc Contains the **nlssrc** command

Examples

1. To get **nlssrc** output in English from a subsystem called **ctsubsys**, enter:

   ```
   nlssrc -c -ls ctsubsys
   ```

2. The following example shows the same information in different formats:

   Local-dependent format:

   ```
   nlssrc -ls ctsubsys

   Subsystem  Group     PID   Status
   ctsubsys   ctsubsys  6334  active
   2 locally-connected clients.  Their PIDs:
   15614 23248
   HA Subsystem domain information:
   Domain established by node 5
   Number of groups known locally: 1
                       Number of          Number of local
   Group Name          providers          providers/subscribers
   ha_filesys             7                   1        0
   ```

   Canonical format:

   ```
   nlssrc -ls ctsubsys -c

   Number of local clients: 2
   PIDs: 15614 23248
   HA Subsystem domain information:
   Domain established by node 5.
   Number of known local groups: 1
   Group Name: ha_filesys
        Providers: 7
        Local Providers: 1
        Local Subscribers: 0
   ```

Related information

Commands: **lssrc**

Chapter 17. Topology services commands

Topology services is a distributed subsystem that provides information to other subsystems about the state of the nodes and adapters in the cluster.

© Copyright IBM Corp. 2002, 2007

cthatsctrl

Purpose

Controls the topology services subsystem.

Syntax

cthatsctrl { **–a** [**-p** *port-number*] | **-s** | **-k** | **-d** | **-b** | **-t** | **-o** | **-r** | **-h** }

Description

The **cthatsctrl** control command controls the operation of the topology services subsystem. The subsystem is under the control of the system resource controller (SRC) and belongs to a subsystem group called **cthats**. Associated with each subsystem is a daemon and a command that configures and starts the daemon.

An instance of the topology services subsystem runs on every node of a cluster.

Adding the subsystem

When the **–a** flag is specified, the control command uses the **mkssys** command to add the topology services subsystem to the SRC. The control command:

1. Makes sure the **cthats** subsystem is stopped.
2. Gets the port number from the cluster data makes sure the port number is set in the **/etc/services** file.

   The service name that is entered in the **/etc/services** file is **cthats**.
3. Removes the **cthats** subsystem from the SRC (in case it is still there).
4. Adds the **cthats** subsystem to the SRC.

Starting the subsystem

When the **–s** flag is specified, the control command uses the **startsrc** command to start to start the topology services subsystem, **cthats**.

Stopping the subsystem

When the **–k** flag is specified, the control command uses the **stoptsrc** command to stop the topology services subsystem, **cthats**.

Deleting the subsystem

When the **–d** flag is specified, the control command uses the **rmssys** command to remove the topology services subsystem from the SRC. The control command:

1. Makes sure the **cthats** subsystem is stopped
2. Removes the **cthats** subsystem from the SRC using the **rmssys** command
3. Removes the **cthats** port number from the **/etc/services** file

Rebuilding the configuration

When the **-b** flag is specified, the control command reads the configuration information from the cluster data and builds a configuration file, **machines.lst**, for the topology services daemon.

Turning tracing on

When the **–t** flag is specified, the control command turns tracing on for the topology services daemon using the **traceson** command.

Turning tracing off

When the **–o** flag is specified, the control command turns tracing off (returns it to its default level) for the topology services daemon using the **tracesoff** command.

Refreshing the subsystem

When the **–r** flag is specified, the control command refreshes the subsystem using the **refresh** command. The **-r** flag signals the daemon to read the rebuilt information.

Flags

–a [–p *port-number***]**
: Adds the subsystem.

-s
: Starts the subsystem.

-k
: Stops the subsystem.

-d
: Deletes the subsystem.

-t
: Turns tracing on for the subsystem.

-o
: Turns tracing off for the subsystem.

-b
: Rebuilds the topology services configuration file from the configuration information in the cluster data.

-r
: Refreshes the subsystem.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

This command writes any error messages to standard error.

Exit status

0
: Indicates that the command completed successfully.

a non-zero value
: Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/cthatsctrl Contains the **cthatsctrl** command

Examples

1. To add the topology services subsystem to the SRC, enter:
   ```
   cthatsctrl -a
   ```
2. To start the topology services subsystem, enter:
   ```
   cthatsctrl -s
   ```
3. To stop the topology services subsystem, enter:
   ```
   cthatsctrl -k
   ```
4. To delete the topology services subsystem from the SRC, enter:
   ```
   cthatsctrl -d
   ```
5. To turn tracing on for the topology services daemon, enter:
   ```
   cthatsctrl -t
   ```
6. To turn tracing off for the topology services daemon, enter:
   ```
   cthatsctrl -o
   ```
7. To rebuild the topology services configuration file from the configuration information in the cluster data, enter:
   ```
   cthatsctrl -b
   ```
8. To signal all the topology services daemons in the cluster to read the new configuration file, enter:
   ```
   cthatsctrl -r
   ```
9. To write usage information to standard output, enter:
   ```
   cthatsctrl -h
   ```

Related information

Commands: **cthactrl**, **cthagsctrl**, **cthatstune**, **lssrc**

cthatstune

Purpose

Views and changes the topology services subsystem's tunable parameters at run time.

Syntax

cthatstune [**-f** [*network1*]**:***frequency1*[**,**[*network2*]**:***frequency2*...]] [**-s** [*network1*]**:***sensitivity1*[**,**[*network2*]**:***sensitivity2*...]] [**-p** *priority*] [**-l** *log_length*] [**-m** *pin_object*] [**-r**] [**-v**] [**-h**]

Description

The **cthatstune** command changes the topology services subsystem's tunable parameters at run time. The topology services subsystem has two types of tunable parameters:

subsystem-wide
Affects the behavior of the topology services subsystem. This type includes the fixed priority level, the maximum length of the log file, and the object to be pinned in main memory.

per-network Affects the behavior of each network. This type includes the heartbeat frequency and sensitivity.

The **cthatstune** command changes the parameters in the cluster data. The new values will not take effect until the topology services daemon reads in the new values from the cluster data. You can use a refresh operation to instruct the topology services daemon to read the new values from the cluster data. You can start a refresh operation by issuing the **cthatsctrl -r** command or the **cthatstune -r** command on one of the nodes in the cluster.

In addition to the real values, two special values: **VIEW** and **DEFAULT**, can be used to display the current setting and to use the default value of the tunable parameter, respectively.

For per-network tunable parameters, in addition to the network name, an empty network name or the special network name **ALL** can be used to specify that the value following the network name applies to all networks.

Flags

-f [*network1***]:***frequency1***[,[***network2***]:***frequency2***...]**
Specifies the *heartbeat frequency*, which is the interval in seconds between heartbeats, for one or more networks.

The value of *frequency* can be an integer from **1** to **30**. The default value is **1**.

-s [*network1***]:***sensitivity1***[,[***network2***]:***sensitivity2***...]**
Specifies the maximum number of missing heartbeats for one or more networks. If this maximum is exceeded, the topology services daemon considers the peer to be inactive.

The value of *sensitivity* can be any integer from **4** to **40**. The default value is **4**.

–p *priority*
Specifies the fixed priority level. The value of *priority* can be **0**, which means “do not run in fixed priority level,” or an integer from **1** to **80**. The default value is **30**.

-l *log_length*
Specifies the maximum log file length (in number of lines). The value of *log_length* can be any integer from **2000** to **1 000 000**. The default value is **5000**.

–m *pin_object* **[,***pin_object***...]**
Specifies the object to be pinned in main memory. Valid values are:

NONE	Does not pin any object in main memory.
TEXT	Specifies the TEXT object to be pinned in main memory.
DATA	Specifies the DATA object to be pinned in main memory.
STACK	Specifies the STACK object to be pinned in main memory.
PROC	Specifies that all pinnable objects should be pinned in main memory. This is the default value.

–r Applies the new tunables and refreshes the topology services subsystem.

–v Provides verbose output.

-h Writes the command’s usage statement to standard output.

Standard output

When the **-h** flag is specified, this command’s usage statement is written to standard output. All verbose messages are written to standard output.

Standard error

This command writes any error messages to standard error.

Exit status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Restrictions

This command is valid in a peer domain only.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/cthatstune Contains the **cthatstune** command

Examples

1. To change the fixed priority level to 40, view the current setting of the maximum log file length, and pin default objects in main memory, without making the new setting take effect immediately, enter:

   ```
   cthatstune -p 40 -l VIEW -m DEFAULT
   ```

2. To make the new setting (previously changed by **cthatstune**) take effect, enter:

   ```
   cthatstune -r
   ```

3. To change the fixed priority level to normal, pin program and data segments in main memory, and make the new settings take effect immediately, enter:

   ```
   cthatstune -p 0 -m TEXT,DATA -r
   ```

4. To change the heartbeat frequency of **filesys_net** to 2 and all other networks to 4, change the sensitivity of all other networks to the default value, and make the new settings take effect immediately, enter:

   ```
   cthatstune -f filesys_net:2,:4 -s :DEFAULT -r
   ```

5. To change the heartbeat frequency of **filesys_net** to the default value and **service_net** to 3, change the sensitivity of all networks to 8, pin the entire topology services subsystem in main memory, and make the new settings take effect immediately, enter:

   ```
   cthatstune -f filesys_net:DEFAULT,service_net:3 -s :8 -m PROC -r
   ```

   You can also do this using the following method:

   ```
   cthatstune -f filesys_net:DEFAULT,service_net:3
   cthatstune -s :8
   cthatstune -m PROC
   cthatstune -r
   ```

Related information

Commands: **cthatsctrl**, **lssrc**

hatsoptions

Purpose

Controls topology services options on a node or a control workstation.

Syntax

hatsoptions [**-s**] [**-d**]

Flags

–s Instructs the topology services daemon to reject messages that are apparently delayed.

–d Instructs the topology services daemon not to reject messages that are apparently delayed (this is the default).

Description

Before this command can be executed, environment variable HB_SERVER_SOCKET must be set to the location of the UNIX-domain socket used by the topology services subsystem. The statement below can be used:

```
export HB_SERVER_SOCKET=/var/ha/soc/hats/server_socket.partition name
```

Alternatively, variable HA_SYSPAR_NAME can be set to the partition name.

The topology services daemon must be running in order for this command to be successful.

hatsoptions can be used to control a number of options in topology services. Option **-s** instructs the topology services daemon to reject messages that are apparently delayed. This can be used in very large system configurations, where messages are sometimes delayed in the network or in the sender and receiver nodes. Use this option only if the Time-Of-Day clocks are synchronized across all the nodes and the control workstation. Otherwise messages may be incorrectly discarded when the sender's Time-Of-Day clock is behind the receiver's.

Option **-d** instructs the topology services daemon not to reject messages that are apparently delayed. This is the default.

Environment variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX-domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to **/var/ha/soc/hats/server_socket.***partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Files

/var/ha/soc/hats/server_socket.*partition name*

Exit status

0 Indicates the successful completion of the command.

1 Indicates the command was unsuccessful.

Security

You must have **root** privilege to run this command.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/hatsoptions
Contains the **hatsoptions** command

Examples

To instruct the topology services daemon on the local node to start discarding apparently delayed messages, enter:

```
export HA_SYSPAR_NAME=partition1

/usr/sbin/rsct/bin/hatsoptions -s
```

Related information

Commands: **hatsctrl**, **hats**, **lssrc**, **startsrc**, **stopsrc**, **syspar_ctrl**

topsvcs

Purpose

Starts or restarts topology services on a cluster node.

Syntax

topsvcs

Description

Use the **topsvcs** script to start the operation of topology services for a cluster.

The **topsvcs** script is not normally run from the command line. It is normally called by the **topsvcsctrl** control script, which is in turn called by the HACMP/ES startup process.

The **topsvcs** script issues these commands:

```
no -o nonlocsrcroute=1
no -o ipsrcroutesend=1
no -o ipsrcrouterecv =1
no -o ipsrcrouteforward=1
```

These commands enable IP source routing. Do not change this setting, because the topology services subsystem requires this setting to work properly. If you change the setting, the topology services subsystem and a number of other subsystems that depend on it will no longer operate properly.

Restrictions

This command is valid in an HACMP™ environment only.

Use this command *only* under the direction of the IBM Support Center.

Implementation specifics

This script is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/topsvcs Contains the **topsvcs** script

Related information

Commands: **lssrc**, **startsrc**, **stopsrc**

Scripts: **topsvcsctrl**

topsvcsctrl

Purpose

Starts the topology services subsystem.

Syntax

topsvcsctrl { **–a** | **–s** | **–k** | **–d** | **–c** | **–u** | **–t** | **–o** | **–r** | **–h** }

Description

The **topsvcsctrl** control script controls the operation of the topology services subsystem. The subsystem is under the control of the system resource controller (SRC) and belongs to a subsystem group called **topsvcs**. This script is normally started by the HACMP/ES startup process.

An instance of the topology services subsystem runs on every node of a cluster.

From an operational point of view, the topology services subsystem group is organized as follows:

Subsystem
topology services

Subsystem group
topsvcs

SRC subsystem
topsvcs

The **topsvcs** subsystem is associated with the **hatsd** daemon and the **topsvcs** script. The **topsvcs** script configures and starts the **hatsd** daemon. The subsystem name on the nodes is **topsvcs**. There is one of each subsystem per node and it is associated with the cluster to which the node belongs.

Daemons
hatsd

Provides the topology services. The **topsvcs** script configures and starts the **hatsd** daemon.

The **topsvcsctrl** script is not normally executed from the command line. It is normally called by the HACMP/ES startup command.

The **topsvcsctrl** script provides a variety of controls for operating the topology services subsystems:

- Adding, starting, stopping, and deleting the subsystems
- Cleaning up the subsystems, that is, deleting them from all system partitions
- Turning tracing on and off
- Refreshing the subsystem

Before performing any of these functions, the script obtains the current cluster name (using the **cllsclstr** command) and the node number (using the **clhandle** command). If the node number is **0**, the control script is running on the control workstation.

Except for the clean and unconfigure functions, all functions are performed within the scope of the current system partition.

Adding the subsystem: When the **–a** flag is specified, the control script uses the **mkssys** command to add the topology services subsystem to the SRC. The control script operates as follows:

1. It makes sure the **topsvcs** subsystem is stopped.
2. It removes the **topsvcs** subsystem from the SRC (in case it is still there).
3. It adds the **topsvcs** subsystem to the SRC.

Starting the subsystem: When the **-s** flag is specified, the control script uses the **startsrc** command to start the topology services subsystem, **topsvcs**.

Stopping the subsystem: When the **-k** flag is specified, the control script uses the **stopsrc** command to stop the topology services subsystem, **topsvcs**.

Deleting the subsystem: When the **-d** flag is specified, the control script uses the **rmssys** command to remove the topology services subsystem from the SRC. The control script operates as follows:

1. It makes sure that the **topsvcs** subsystem is stopped.
2. It removes the **topsvcs** subsystem from the SRC using the **rmssys** command.
3. It removes the port number from the **/etc/services** file.

Cleaning up the subsystems: When the **-c** flag is specified, the control script stops and removes the topology services subsystems for all clusters partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the clusters, using the **stopsrc -g topsvcs** command.
2. It removes all entries for the **topsvcs** subsystem from the **/etc/services** file.

Turning tracing on: When the **-t** flag is specified, the control script turns tracing on for the **hatsd** daemon, using the **traceson** command.

Turning tracing off: When the **-o** flag is specified, the control script turns tracing off (returns it to its default level) for the **hatsd** daemon, using the **tracesoff** command.

Refreshing the subsystem: When the **-r** flag is specified, the control script refreshes the subsystem, using the **topsvcs refresh** command and the **refresh** command. It rebuilds the information about the node and adapter configuration in the global object data manager (ODM) and signals the daemon to read the rebuilt information.

Logging: While it is running, the topology services daemon (**hatsd**) provides information about its operation and errors by writing entries in a log file called **/var/ha/log/topsvcs.***cluster_name*.

Flags

-a Adds the subsystem.

-s Starts the subsystem.

-k Stops the subsystem.

-d Deletes the subsystem.

-c Cleans the subsystems.

-u Removes the topology services subsystem from all partitions.

-t Turns tracing on for the subsystem.

-o Turns tracing off for the subsystem.

-r Refreshes the subsystem.

-h Writes the script's usage statement to standard output.

Files

/var/ha/log/topsvcs.*cluster_name*
Contains the log of the **hatsd** daemon on the cluster named **cluster_name**

Standard error

This script writes error messages (as necessary) to standard error.

Exit status

0 Indicates that the script completed successfully.

1 Indicates that an error occurred.

Security

You must be running with an effective user ID of **root** to use this script.

Restrictions

This command is valid in an HACMP environment only.

Use this command *only* under the direction of the IBM Support Center.

Implementation specifics

This script is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/topsvcsctrl Contains the **topsvcsctrl** script

Examples

1. To add the topology services subsystem to the SRC, enter:
   ```
   topsvcsctrl -a
   ```
2. To start the topology services subsystem, enter:
   ```
   topsvcsctrl -s
   ```
3. To stop the topology services subsystem, enter:
   ```
   topsvcsctrl -k
   ```
4. To delete the topology services subsystem from the SRC, enter:
   ```
   topsvcsctrl -d
   ```
5. To clean up the topology services subsystem, enter:
   ```
   topsvcsctrl -c
   ```
6. To turn tracing on for the topology services daemon, enter:

```
topsvcsctrl -t
```

7. To turn tracing off for the topology services daemon, enter:

```
topsvcsctrl -o
```

Related information

Commands: **lssrc**, **startsrc**, **stopsrc**

Scripts: **topsvcs**

Chapter 18. Group services commands

Group services provides distributed coordination and synchronization services for other distributed subsystems running on a set of nodes in a cluster.

© Copyright IBM Corp. 2002, 2007

cthagsctrl

Purpose

Controls the group services subsystem.

Syntax

cthagsctrl { **-a** [**-p** *port-number*] | **-d** | **-s** | **-k** | **-t** | **-o** | **-r** | **-z** | **-h** }

Description

The **cthagsctrl** control command controls the operation of the group services subsystem (**cthags**) under the control of the system resource controller (SRC).

An instance of the group services subsystem runs on every node of a cluster.

From an operational point of view, the group services subsystem group is organized as follows:

Subsystem	group services
Subsystem group	**cthags**
SRC subsystem	**cthags** — Associated with the **hagsd** daemon. The subsystem name on the nodes is **cthags**. There is one subsystem per node and each of these subsystems is associated with the cluster to which the node belongs.
Daemon	**hagsd** — Provides the group services functions.

In general, the **cthagsctrl** command is not issued from the command line. It is normally called by the **cthactrl** command during the creation of the cluster.

The **cthagsctrl** command provides a variety of controls for operating the group services subsystem:

- Adding, deleting, starting, and stopping the subsystem
- Deleting the subsystem from the cluster
- Turning tracing on and off
- Refreshing the subsystem configuration within the cluster

Adding the subsystem

When the **-a** flag is specified, the **cthagsctrl** command adds the group services subsystems to the SRC. This command:

1. Makes sure the **cthags** subsystem is stopped.
2. Gets the port number for the **cthags** subsystem from the cluster data.
3. Removes the **cthags** subsystem from the SRC (in case it is still there).
4. Adds the **cthags** subsystem to the SRC.
5. Does not currently add an entry for the **cthags** group to the **/etc/inittab** file. As a result, **cthags** is required to be started by another subsystem when it is needed.

Deleting the subsystem

When the **–d** flag is specified, this command uses the **rmssys** command to remove the group services subsystems from the SRC. The control command:

1. Makes sure the **cthags** subsystem is stopped.
2. Removes the **cthags** subsystem from the SRC using the **rmssys** command.
3. Removes the port number from the **/etc/services** file.

Starting the subsystem

When the **–s** flag is specified, this command uses the **startsrc** command to start the group services subsystem, **cthags**.

Stopping the subsystem

When the **–k** flag is specified, this command uses the **stopsrc** command to stop the group services subsystem, **cthags**.

Turning tracing on

When the **–t** flag is specified, this command uses the **traceson** command to turn tracing on for the **hagsd** daemon.

Turning tracing off

When the **–o** flag is specified, this command uses the **tracesoff** command to turn tracing off (returns it to its default level) for the **hagsd** daemon.

Refreshing the subsystem

The **–r** flag refreshes the **cthags** subsystem.

Logging

While they are running, the group services daemons provide information about their operation and errors by writing entries in three log files in the **/var/ct/***cluster_name***/log/cthags** directory. The log files are:

- **/var/ct/***cluster_name***/log/cthags**_*nodenum_instnum***.***cluster_name*
- **/var/ct/***cluster_name***/log/cthags**_*nodenum_instnum.cluster_name***.long**
- **/var/ct/***cluster_name***/log/cthags.default.***nodenum_instnum*

The log files contain the log of the **hagsd** daemons on the nodes.

The log file names include these variables:

- *nodenum* is the node number on which the daemon is running.
- *instnum* is the instance number of the daemon.
- *cluster_name* is the name of the cluster in which the daemon is running.

Each daemon limits the log size to a pre-established number of lines. The default is 5000 lines. When the limit is reached, the daemon appends the string **.bak** to the name of the current log file and begins a new log. If a **.bak** version already exists, it is removed before the current log is renamed.

Flags

–a [–p *port number* **]**
Adds the subsystem.

-d Deletes the subsystem.

-s Starts the subsystem.

-k Stops the subsystem.

-t Turns tracing on for the subsystem.

-o Turns tracing off for the subsystem.

-r Refreshes the subsystem.

-z Uninstalls the **cthags** subsystem.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

This command writes error messages, as necessary, to standard error.

Exit status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/cthagsctrl Contains the **cthagsctrl** command

Examples

1. To add the group services subsystems to the SRC in the current cluster, enter:
   ```
   cthagsctrl -a
   ```
2. To add the group services subsystems with a port number of 12347, enter:
   ```
   cthagsctrl -a -p 12347
   ```

3. To delete the group services subsystems from the SRC in the current cluster, enter:
   ```
   cthagsctrl -d
   ```
4. To start the group services subsystems in the current cluster, enter:
   ```
   cthagsctrl -s
   ```
5. To stop the group servicess subsystems in the current cluster, enter:
   ```
   cthagsctrl -k
   ```
6. To turn tracing on for the group services daemon in the current cluster, enter:
   ```
   cthagsctrl -t
   ```
7. To turn tracing off for the group services daemon in the current cluster, enter:
   ```
   cthagsctrl -o
   ```

Related information

Commands: **cthactrl**, **cthagstune**, **lssrc**, **nlssrc**

cthagstune

Purpose

Changes the group services subsystem's tunable parameters at run time.

Syntax

cthagstune { [**-l** *log_length*] | [**-d** *log_dirsize*] | [**-h**] }

Description

The **cthagstune** command changes the group services subsystem's tunable parameters at run time.

Flags

-l Specifies the maximum log file length. If the value is **0** or a negative number, a default log file length is used.

-d Specifies the maximum log directory size in kilobytes. If the value is **0** or a negative number, a default log directory size is used.

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

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

This command writes error messages, as necessary, to standard error.

Exit status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Restrictions

This command is valid in a peer domain only.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/cthagstune Contains the **cthagstune** command

Examples

To change the log file length to 6000 lines and to set the log directory size to approximately 7 megabytes, enter:

```
cthagstune -l 6000 -d 7000
```

Related information

Commands: **cthactrl**, **cthagsctrl**, **lssrc**, **nlssrc**

grpsvcsctrl

Purpose

Starts the group services subsystems.

Syntax

grpsvcsctrl { **–a** | **–s** | **–k** | **–d** | **–c** | **–u** | **–t** | **–o** | **–h** }

Description

The **grpsvcsctrl** command starts the group services subsystems. This control script controls the operation of the subsystems that are required for group services. These subsystems are under the control of the system resource controller (SRC) and belong to a subsystem group called **grpsvcs**. A daemon is associated with each subsystem. From an operational point of view, the group services subsystem group is organized as follows:

Subsystem	group services
Subsystem group	**grpsvcs**
SRC subsystem	**grpsvcs** — associated with the **hagsd** daemon. The subsystem name on the nodes is **grpsvcs**. The **grpsvcs** subsystem on each node is associated with the cluster to which the node belongs.
Daemon	**hagsd** — provides the majority of the group services functions.

The **grpsvcsctrl** script is not normally run from the command line. It is normally called by the startup command during installation of the cluster.

The **grpsvcsctrl** script provides a variety of controls for operating the group services subsystems:

- Adding, starting, stopping, deleting, and cleaning up the subsystems
- Turning tracing on and off

Before performing any of these functions, the script obtains the current cluster name.

Adding the subsystem: When the **-a** flag is specified, the control script uses the **mkssys** command to add the group services subsystems to the SRC. The control script operates as follows:

1. It makes sure the **grpsvcs** subsystem is stopped.
2. It gets the port number for the **grpsvcs** subsystem for this cluster from the global object data manager (ODM) and makes sure the port number is set in the **/etc/services** file. The range of valid port numbers is 10000 to 10100, inclusive.
3. The service name that is entered in the **/etc/services** file is **grpsvcs.***cluster_name*.
4. It removes the **grpsvcs** subsystem from the SRC (in case it is still there).
5. It adds the **grpsvcs** subsystem to the SRC. The cluster name is configured as a daemon parameter on the **mkssys** command.

Starting the subsystem: When the **-s** flag is specified, the control script uses the **startsrc** command to start the group services subsystem, **grpsvcs**.

Stopping the subsystem: When the **-k** flag is specified, the control script uses the **stopsrc** command to stop the group services subsystem, **grpsvcs**.

Deleting the subsystem: When the **-d** flag is specified, the control script uses the **rmssys** command to remove the group services subsystem from the SRC. The control script operates as follows:

1. It makes sure the **grpsvcs** subsystem is stopped.
2. It removes the **grpsvcs** subsystem from the SRC using the **rmssys** command.
3. It removes the port number from the **/etc/services file**.

Cleaning up the subsystems: When the **-c** flag is specified, the control script stops and removes the group services subsystems for all system partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the subsystem group in all partitions, using the **stopsrc -g grpsvcs** command.
2. It removes all instances of subsystems in the subsystem group in all partitions from the SRC using the **rmssys** command.

Turning tracing on: When the **-t** flag is specified, the control script turns tracing on for the **hagsd** daemon, using the **traceson** command.

Turning tracing off: When the **-o** flag is specified, the control script turns tracing off (returns it to its default level) for the **hagsd** daemon, using the **tracesoff** command.

Logging: While they are running, the group services daemons provide information about their operation and errors by writing entries in a log file in the **/var/ha/log** directory.

Each daemon limits the log size to a pre-established number of lines. The default is 5000 lines. When the limit is reached, the daemon appends the string **.bak** to the name of the current log file and begins a new log. If a **.bak** version already exists, it is removed before the current log is renamed.

Flags

-a Adds the subsystem.

-s Starts the subsystems.

-k Stops the subsystems.

-d Deletes the subsystems.

-c Cleans the subsystems (that is, deletes them from all system partitions).

-u Removes the group services subsystem from all partitions.

-t Turns tracing on for the subsystems.

-o Turns tracing off for the subsystems.

-h Writes the script's usage statement to standard output.

Files

/var/ha/log/grpsvcs_nodenum_instnum.cluster_name
Contains the log of the **hagsd** daemons on the nodes

The file name includes these variables:

nodenum
is the node number on which the daemon is running

instnum
is the instance number of the daemon

cluster_name
is the name of the cluster in which the daemon is running

Standard error

This command writes error messages (as necessary) to standard error.

Exit status

0 Indicates the successful completion of the command.

1 Indicates that an error occurred.

Security

You must be running with an effective user ID of **root**.

Restrictions

This script is valid in an HACMP environment only.

Implementation specifics

This script is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/grpsvcsctrl Contains the **grpsvcsctrl** script

Examples

1. To add the group services subsystems to the SRC, enter:
   ```
   grpsvcsctrl -a
   ```
2. To start the group services subsystems, enter:
   ```
   grpsvcsctrl -s
   ```
3. To stop the group services subsystems, enter:
   ```
   grpsvcsctrl -k
   ```
4. To delete the group services subsystems from the SRC, enter:
   ```
   grpsvcsctrl -d
   ```
5. To clean up the group services subsystems, enter:
   ```
   grpsvcsctrl -c
   ```
6. To turn tracing on for the group services daemon **hagsd**, enter:
   ```
   grpsvcsctrl -t
   ```

7. To turn tracing off for the group services daemon **hagsd**, enter:
   ```
   grpsvcsctrl -o
   ```

Related information

Commands: **lssrc**, **mkssys**, **rmssys**, **startsrc**, **stopsrc**

Daemons: **hagsd**

hagsd

Purpose

Observes resource variable instances that are updated by resource monitors and generates and reports events to client programs.

Syntax

hagsd *daemon_name*

Parameters

daemon_name
Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Description

The **hagsd** daemon is part of the group services subsystem, which provides a general-purpose facility for coordinating and monitoring changes to the state of an application that is running on the nodes of a cluster. This daemon provides most of the services of the subsystem. *daemon_name* specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

One instance of the **hagsd** daemon executes on each cluster node. The **hagsd** daemon is under the control of the system resource controller (SRC).

Because the daemon is under SRC control, it is better not to start it directly from the command line. It is normally called by the **grpsvcsctrl** command, which is in turn called by the cluster startup process. If you must start or stop the daemon directly, use the **startsrc** or **stopsrc** command.

Implementation specifics

This daemon is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/hagsd Contains the **hagsd** daemon

Related information

Commands: **grpsvcsctrl**

hagsns

Purpose

Gets group services name server information.

Syntax

hagsns [**–h** *host*] [**–c**] **–s** *subsystem_name*

hagsns [**–h** *host*] [**–c**] **–p** *subsystem_pid*

Description

Use the **hagsns** command to query the status of the group services nameserver.

Flags

–c Forces the output as ″English_only.″ If the **–c** flag is not specified, the daemon's locale will be used for the output.

–h *host*
Specifies the host to obtain name server status for.

–p *subsystem_pid*
Specifies a particular instance of the *subsystem_pid* to obtain name server status for.

–s *subsystem_name*
Specifies a subsystem to get status for. The *subsystem_name* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* variable is not contained in the subsystem object class.

Standard error

This command writes error messages, as necessary, to standard error.

Exit status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/hagsns Contains the **hagsns** command

Examples

To get domain information from the group services subsystem, enter:

```
hagsns -c -s cthags
```

or

```
hagsns -s cthags
```

The output will look like this:

```
HA GS NameServer Status
NodeID=1.16, pid=14460, domainID=6.14, NS established,CodeLevel=GSLevel(DRL=8)
NS state=kCertain, protocolInProgress=kNoProtocol,outstandingBroadcast=KNoBcast
Process started on Jun 19 18:34:20, (10d 20:19:22) ago, HB connection took (19:14:9).
Initial NS certainty on Jun 20 13:48:45, (10d 1:4:57) ago, taking (0:0:15).
Our current epoch of Jun 23 13:05:19 started on (7d 1:48:23), ago.
Number of UP nodes: 12
List of UP nodes: 0 1 5 6 7 8 9 11 17 19 23 26
```

In this example, **domainID=6.14** means that node 6 is the name server (NS) node. The domain ID consists of a node number and an incarnation number. The incarnation number is an integer, incremented whenever the group services daemon is started. **NS established** means that the name server was established.

Related information

Commands: **hagsvote**, **lssrc**, **nlssrc**

hagsvote

Purpose

Gets vote information for group services groups.

Syntax

hagsvote [**–h** *host*] [**–l**] [**–a** *argument*] [**–c**] **–s** *subsystem_name*

hagsvote [**–h** *host*] [**–l**] [**–a** *argument*] [**–c**] **–p** *subsystem_pid*

Description

Use the **hagsvote** command to query the status of voting protocols for group services.

Flags

–a *argument*
: Specifies the name of a group services group that was created from the client's first call to join the protocol.

–c
: Requests the canonical output of the group services voting information. The output is displayed in English regardless of the installed language locale. If **–c** is not specified, the daemon's locale will be used for the output.

–h *host*
: Specifies the host name which is getting status.

–l
: Requests detailed output in "long" form.

–p *subsystem_pid*
: Specifies a particular instance of the *subsystem_pid* variable to get the vote for.

–s *subsystem_name*
: Specifies a subsystem to vote on. The *subsystem_name* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* variable is not contained in the subsystem object class.

Standard output

This command writes error messages (as necessary) to standard error.

Exit status

0
: Indicates the successful completion of the command.

a non-zero value
: Indicates that an error occurred.

Security

You must have **root** privilege to run this command.

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/hagsvote Contains the **hagsvote** command

Examples

1. To see information about the status of the voting protocol for the group **theSourceGroup** in long form, enter:

   ```
   hagsvote -ls cthags -a theSourceGroup (locale-dependent)
   ```

   The output will look like this:

   ```
   Number of groups: 4
   Group name [theSourceGroup] GL node [26] voting data:
   GL in phase [1] of n-phase protocol of type [Join].
   Local voting data:
   Number of providers: 1
   Number of providers not yet voted: 1 (vote not submitted).
   Given vote: [No vote value] Default vote: [No vote value]
   ProviderID Voted? Failed? Conditional?
   [101/26] No No Yes
   Global voting data:
   Number providers not yet voted: 1
   Given vote: [No vote value] Default vote: [No vote value]
   Nodes that have voted: []
   Nodes that have not voted: [26]
   ```

   The first line of the output means that the total number of groups is 4. The second line provides the group name and the group leader node (in this case 26). The remaining lines give the voting data:

   - The group leader is in phase 1 of an *n*-phase protocol.
   - The protocol is the **Join** protocol.
   - For the local node, it has one provider. One provider has not voted yet.
   - No default vote value is given and no vote value is given.
   - Under the line ″ProviderID Voted? Failed? Conditional?,″ ″[101/16] No No Yes,″ means that the provider ID is 101/26, not voted yet, not failed, but wait for the vote (so it is conditional).

   The output then shows the global voting status:

   - One provider has not voted yet.
   - No vote value given yet, no default vote value.
   - The nodes that have voted is none.
   - The nodes that have not voted is node 26.

2. In the following example, the meaning of each line of output is the same as in the first example except that node 26 is the group leader node.

   ```
   hagsvote -ls cthags -a theSourceGroup -c (canonical form)
   ```

   The output will look like this:

   ```
   Number of groups: 4
   Group Name: theSourceGroup
   GL Node: 26 (I am GL)
   Current phase number of an n-phase protocol: 1
   Protocol name: [Join]
   Local voting data:
   Number of local providers: 1
   Number of local providers not yet voted: 1 (vote not submitted)
   Given vote: [No vote value] Default vote: [No vote value]Global voting data:
   Number of nodes in group: 1
   Number of global providers not yet voted: 1
   Given vote: [No vote value] Default vote: [No vote value]
   Nodes that have voted: []
   Nodes that have not voted: [26]
   ```

Related information

Commands: **hagsns**, **lssrc**, **nlssrc**

Part 9. Virtual shared disk

Chapter 19. Virtual shared disk commands 473
cfgvsd 474
createvsd 476
ctlvsd 482
defvsd 485
fencevsd 487
ha_vsd 488
ha.vsd 489
lsvsd 493
preparevsd 496
removevsd 498
resumevsd 500
rvsdrestrict 502
startvsd 504
statvsd 506
stopvsd 508
suspendvsd 510
ucfgvsd 512
undefvsd 514
unfencevsd 515
updatevsdnode 517
updatevsdtab 520
updatevsdvg 521
vsdatalst 524
vsdchgserver 526
vsdelnode 528
vsdelvg 530
vsdnode 532
vsdsklst 535
vsdvg 537
vsdvgts 539

© Copyright IBM Corp. 2002, 2007

Chapter 19. Virtual shared disk commands

© Copyright IBM Corp. 2002, 2007

cfgvsd

Purpose

Configures a virtual shared disk.

Syntax

cfgvsd {**–a** | *vsd_name* ...}

Description

Use this command to configure the already defined virtual shared disks and bring them to the stopped state. This command does not make the virtual shared disk available.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

You can use the System Management Interface Tool (SMIT) to run the **cfgvsd** command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Configure a virtual shared disk** option.

Parameters

vsd_name Specifies a defined virtual shared disk.

Flags

–a Specifies all virtual shared disks that have been defined.

Security

You must have **root** authority to run this command.

Restrictions

Under normal circumstances, you should not issue this command. The RVSD subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/cfgvsd

Examples

To bring the virtual shared disk **vsd1vg1n1** from the defined state to the stopped state, enter:

```
cfgvsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **ctlvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startrpdomain**, **startrpnode**, **startvsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

createvsd

Purpose

Creates a set of virtual shared disks, with their associated logical volumes.

Syntax

createvsd **–n** {*node_list* | **ALL**} **–s** *size_in_MB* **–g** *vg_name*
[{**–c** *vsds_per_node* | **–L**}] [–A]
[{**–m** *mirror_count* | **–p** *lvm_stripe_size_in_K*}] [**–v** *vsd_name_prefix*]
[**–l** *lv_name_prefix*] [**–T** *lp_size_in_MB*] [**–k** *vsd_type*] [**–x**]

Description

Use this command to create a volume group with the specified name (if one does not already exist) and to create a logical volume within that volume group. You specify the logical volume size using the **-s** flag.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_data
```

and select the **Create a virtual shared disk** option.

Flags

Note: Some examples shown in this list do not contain enough flags to be executable. They are shown in an incomplete form to illustrate specific flags.

–n *node_list* Specifies the node numbers of the peer domain on which you are creating virtual shared disks. The backup node cannot be the same as the primary node. For nonconcurrent virtual shared disks, the format of the node list is:

```
[P/S] : disk_list1+disk_list2/
```

For concurrent virtual shared disks, the format of the node list is:

```
[S1/S2/...Sn] : disk_list1+disk_list2/
```

″P″ specifies the primary server node for serially accessed shared disks, ″S″ specifies the backup (secondary) server node for serially accessed shared disks, and S1 and S2 specifies the server nodes for concurrently accessed shared disks. *disk_list1* is the list of local physical disks, or vpaths, for the logical volume on the primary. In other words, this list can be made up of *hdiskx, hdisky,...* or *vpathx, vpathy,....*

Notes:

1. Vpaths are available only if the ″Subsystem Device Driver″ is installed. Vpaths provide ″virtual paths″ to the same physical volume.
2. Hdisks and vpaths cannot both be specified in the same list.

disk_list1+disk_list2 is the list of local physical disks or vpaths in the volume group on the primary, if you want to have more disks in the volume group than are needed for the logical volume. The sequence in which nodes are listed determines the names given to the virtual shared disks. For example:

```
createvsd -n 1,6,4 -v PRE
```

(with the *vsd_prefix* PRE) creates virtual shared disks PRE1n1 on node 1, PRE2n6 on node 6, and PRE3n4 on node 4.

To create a volume group that spans hdisk2, hdisk3, and hdisk4 on node 1, with a backup on node 3, enter:

```
createvsd -n 1/3:hdisk2,hdisk3,hdisk4/ -v DATA
```

This command creates:

- Virtual shared disk DATA1n1 with logical volume lvDATA1n1 on a volume group with the global volume group name DATA1n1b3 on node 1, exported to node 3. The Logical Volume Manager (LVM) volume group name is DATA. The logical volumes span hdisk2, hdisk3, and hdisk4.

To create volume groups just like that one on nodes 1, 2, and 3 of a system with backup on nodes 4, 5, and 6 of the same system, enter:

```
createvsd -n 1/4:hdisk1,hdisk2,hdisk3/,2/5:hdisk5,hdisk6, \
          hdisk7/,3/6:hdisk2,hdisk4,hdisk6/ -v DATA
```

This command is shown on two lines here, but you must enter it without any spaces between the items in *node_list*.

The command creates:

- Virtual shared disk DATA1n1 with logical volume lvDATA1n1 on a volume group with the local volume group name DATA on node 1, exported to node 4. The global volume group name is DATAn1b4.
- Virtual shared disk DATA2n2 with logical volume lvDATA2n2 on a volume group with the local volume group name DATA on node 2, exported to node 5. The global volume group name is DATAn2b5.
- Virtual shared disk DATA3n3 with logical volume lvDATA3n3 on a volume group with the local volume group name DATA on node 3, exported to node 6. The global volume group name is DATAn3b6.

To create a virtual shared disk where the logical volume spans only two of the physical disks in the volume group, enter:

```
createvsd -n 1/3:hdisk1,hdisk2+hdisk3/ -v DATA
```

This command creates the virtual shared disk DATA1n1 with logical volume lvDATA1n1 spanning hdisk1 and hdisk2 in the volume group DATA, which includes hdisk1, hdisk2, and hdisk3. It exports the volume group DATA to node 3.

If a volume group is already created and the combined physical hdisk lists contain disks that are not needed for the logical volume,

those hdisks are added to the volume group. If the volume group has not already been created, **createvsd** creates a volume group that spans *hdisk_list1+hdisk_list2*.

Backup nodes cannot use the same physical disk as the primary does to serve virtual shared disks.

ALL specifies that you are creating virtual shared disks on all nodes in the RSCT peer domain. No backup nodes are assigned if you use this operand. The virtual shared disks will be created on all the physical disks attached to the nodes in *node_list* (you cannot specify which physical disks to use.)

–s Specifies the size in megabytes of each virtual shared disk.

–g Specifies the Logical Volume Manager (LVM) volume group name. This name is concatenated with the node number to produce the global volume group name. For example:

```
createvsd -n 6 -g VSDVG
```

creates a volume group with the local volume group name VSDVG and the global volume group name VSDVG1n6 on node 6. The node number is added to the prefix to avoid name conflicts when a backup node takes over a volume group. If a backup node exists, the global volume group name will be concatenated with the backup node number as well as the primary. For example:

```
createvsd -n 6/3/ -g VSDVG
```

creates a volume group with the local volume group name VSDVG and the global volume group name VSDVGn6b3. The primary node is node 6 and the backup node for this volume group is node 3.

–c Specifies the number of virtual shared disks to be created on each node. If *number_of_vsds_per_node* is not specified, one virtual shared disk is created for each node specified on **createvsd**. If more than one virtual shared disk is to be created for each node, the names will be allocated alternately. For example:

```
createvsd -n 1,6 -c 2 -v DATA
```

creates virtual shared disks DATA1n1 on node 1, DATA2n6 on node 6, DATA3n1 on node 1, and DATA4n6 on node 6.

–L Allows you to create one virtual shared disk on each node without using sequential numbers, for locally accessed virtual shared disks.

–A Specifies that virtual shared disk names will be allocated to each node in turn, for example:

```
createvsd -n 1,6 -c 2 -A DATA
```

creates DATA1n1 and DATA2n1 on node 1, and DATA3n6 and DATA4n6 on node 6.

–m Specifies the LVM mirroring count. The mirroring count sets the number of physical partitions allocated to each logical partition. The range is from 1 to 3 and the default value is 1.

–p Specifies the LVM stripe size. If this flag is not specified, the logical volumes are not striped. To use striping, the node on which the virtual shared disks are defined must have more than one physical disk.

–v Specifies a prefix to be given to the names of the created virtual shared disks. This prefix will be concatenated with the virtual shared disk number, node number, and backup node number, if a backup disk is specified. For example, if the prefix PRE is given to a virtual shared disk created on node 1 and there are already two virtual shared disks with this prefix across the partition, the new virtual shared disk name will be PRE3n1. The name given to the underlying logical volume will be lvPRE3n1, unless the **–l** flag is used. The **createvsd** command continues to sequence virtual shared disk names from the last PRE-prefixed virtual shared disk.

If **-v** is not specified, the prefix `vsd` is used.

Note: The last character of the *vsd_name_prefix* cannot be a digit. Otherwise, the 11th virtual shared disk with the prefix PRE would have the same name as the first virtual shared disk with the prefix PRE1. Nor can the *vsd_name_prefix* contain the character '**.**' because '**.**' can be any character in regular expressions.

–l Overrides the prefix *lvx* that is given by default to a logical volume by the **createvsd** command, where *x* is the virtual shared disk name prefix specified by *vsd_name_prefix* or the default (`vsd`). For example:

```
createvsd -n 1 -v DATA
```

creates one virtual shared disk on node 1 named DATA1n1 with an underlying logical volume lvDATA1n1. If the command

```
createvsd -n 1 -v DATA -l new
```

is used, the virtual shared disk on node 1 is still named DATA1n1, but the underlying logical volume is named lvnew1n1.

It is usually more helpful **not** to specify **–l**, so that your lists of virtual shared disk names and logical volume names are easy to associate with each other and you avoid naming conflicts.

–T Specifies the size of the physical partition in the Logical Volume Manager (LVM) logical volume group and also the logical partition size (they will be the same) in megabytes. You must select a power of 2 in the range 2 – 256. The default is 4MB.

The Logical Volume Manager limits the number of physical partitions to 1016 per disk. If a disk is greater than 4 gigabytes in size, the physical partition size must be greater than 4MB to keep the number of partitions under the limit.

–k *vsd_type* Specifies the type of virtual shared disk. The options are:

- **VSD**: specifies a serial access, or nonconcurrent, shared disk, or
- **CVSD**: specifies a concurrent access shared disk.

The default is **VSD**.

–x Specifies that the steps required to synchronize the virtual shared disks on the primary and secondary nodes should **not** be performed; that is, the sequence:

- **varyoffvg** on the primary node
- **exportvg** on the secondary node

- **importvg** on the secondary node
- **chvg** on the secondary node
- **varyoffvg** on the secondary node
- **varyonvg** on the primary nodes

is not done as part of the **createvsd** processing. This speeds the operation of the command and avoids unnecessary processing in the case where several virtual shared disks are being created on the same primary/secondary nodes. In this case, however, you should either **not** specify **–x** on the last **createvsd** in the sequence or issue the volume group commands listed above explicitly.

Standard output

For the following command:

```
createvsd -n 1/2:hdisk13/ -s 1024 -g testvg -v testvsd -T 16
```

The messages to standard output will be similar to:

```
createvsd: calls Getopts.
createvsd: parsing node_list.
createvsd: creates task tables.
createvsd: calls checkclvm.perl on the nodes c164n11.ppd.pok.ibm.com
createvsd: calls domkvglv.perl.
OK:1:mkvg -f -y testvg -s 16 hdisk13
OK:1:mklv -a c -y lvtestvsd1n1 -e x testvg 64 hdisk13
It took about 8 seconds in mkvglv.
createvsd: calls dovaryoffvg.perl testvg on the primary node c164n11.ppd.pok.ibm.com
OK:1:chvg -a n testvg
OK:1:varyoffvg testvg
createvsd: calls doimportvg.perl testvg on the nodes c164n12.ppd.pok.ibm.com with 000048186b991a6f
importvg : testvg
importvg : OK:2:importvg -y testvg hdisk5
importvg : OK:2:chvg -a n testvg
importvg : timestamp 2 testvg 3e036cb33403c8c8
importvg : OK:2:varyoffvg testvg
importvg : It took about 10 seconds.
It took about 12 seconds in importvg.
createvsd: calls vsdvg.
OK:1:vsdvg -g testvgn1b2 testvg 1 2
It took about 12 seconds in vsdvg.
createvsd: calls dovaryonvg.perl testvg on pri nodes c164n11.ppd.pok.ibm.com
OK:1:varyonvg  testvg
createvsd: calls defvsd.
OK:1:defvsd lvtestvsd1n1 testvgn1b2 testvsd1n1
It took about 5 seconds in defvsd.
```

Exit status

0 Indicates the successful completion of the command.

–1 Indicates that an error occurred.

Security

You must have **root** authority to run this command.

Restrictions

1. The backup node cannot be the same as the primary node.
2. The last character of *vsd_name_prefix* cannot be numeric.
3. The *vsd_name_prefix* cannot contain the character '**.**'.

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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/createvsd

Examples

To create two 4MB virtual shared disks on each of three primary nodes, one of which has a backup, enter:

```
createvsd -n 3,4,7/8/ -c 2 -s 4 -g vsdvg -v TEMP
```

This command creates the following virtual shared disks:

- TEMP1n3, with logical volume lvTEMP1n3 on a volume group with the global volume group name vsdvgn3 on node 3
- TEMP2n4, with logical volume lvTEMP2n4 on a volume group with the global volume group name vsdvgn4 on node 4
- TEMP3n7, with logical volume lvTEMP3n7 on a volume group with the global volume group name vsdvgn7b8 on node 7, also imported to node 8
- TEMP4n3, with logical volume lvTEMP4n3 on a volume group with the global volume group name vsdvgn3 on node 3
- TEMP5n4, with logical volume lvTEMP5n4 on a volume group with the global volume group name vsdvgn4 on node 4
- TEMP6n7, with logical volume lvTEMP6n7 on a volume group with the global volume group name vsdvgn7b8 on node 7, also imported to node 8

To create three virtual shared disks, where the logical volume created on node 3 spans fewer disks than the volume group does, enter:

```
createvsd -n 3,4/:hdisk1,hdisk2+hdisk3/,7/8/ -s 4 -g datavg -v USER
```

This command creates:

- USER1n3, with logical volume lvUSER1n3 defined on a volume group with the global volume group name datavgn3 on node 3.
- USER2n4, with logical volume lvUSER2n4 defined on a volume group with the global volume group name datavgn4 on node 4. datavgn4 spans hdisk1, hdisk2, and hdisk3. lvUSER2n4 spans hdisk1 and hdisk2.
- USER3n7, with logical volume lvUSER3n7 defined on a volume group with the global volume group name datavgn7b8 on node 7, also imported to node 8.
- If no volume group was defined on nodes 3 and 7 before this **createvsd** command was issued, the volume groups datavgn3 and datavgn7b8 are created with one 4MB partition from a single physical disk.

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **defvsd**, **startrpdomain**, **startrpnode**, **vsdvg**

ctlvsd

Purpose

Sets the operational parameters for the virtual shared disk subsystem on a node.

Syntax

ctlvsd [**–r** *node_number*... | **–R** | **–p** *parallelism* |
–k *node_number*... | **–t** | **–T** | **–v** *vsd_name* ... |
–V | **–C** | **–K** | **–M** *IP_max_message_size*]

Description

The **ctlvsd** command changes some parameters of the virtual shared disk subsystem. When called with no arguments, the command displays the current and maximum cache buffer count, the request block count, the pbuf count, the minimum buddy buffer size, the maximum buddy buffer size, and the overall size of the buddy buffer.

Sequence number information may or may not be displayed. In general, sequence numbers and the options that reset them are managed entirely within the virtual shared disk and recoverable virtual shared disk subsystems.

Flags

–r Resets the outgoing and expected sequence numbers for the nodes specified on the node on which the command is run. Use this flag when another node has either been rebooted, cast out, or all virtual shared disks have been reconfigured on that node. The specified nodes are also cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

–R Resets the outgoing and expected sequence number for all nodes on the node on which the command is run. Use this flag after rebooting the node. All nodes in the virtual shared disk network will be cast in.

Note: This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

–p Sets the level of virtual shared disk parallelism to the number specified. The valid range is 1 to 9. The default is 9. A larger value can potentially give better response time to large requests. (See *RSCT for AIX: Managing Shared Disks* for more information regarding tuning virtual shared disk performance.)

This value is the *buf_cnt* parameter on the **uphysio** call that the virtual shared disk IP device driver makes in the kernel. Use **statvsd** to display the current value on the node on which the command is run.

–k Casts out the node numbers specified on the local node. The local node ignores requests from cast out nodes. Use **–r** to cast nodes back in.

Notes:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

–t Lists the current routing table and mbuf headers cached by the virtual shared disk driver.

–T Clears or releases all cached routes.

–v *vsd_name* **...**
Resets the statistics in the number of read and write requests on the specified virtual shared disks.

–V Resets all the configured virtual shared disk’s statistics in the number of read and write requests.

–C Resets the virtual shared disk device driver counters displayed by the **statvsd** command. Exceptions are the outgoing and expected request sequence numbers among the client and server nodes.

–K Casts out all nodes on the local node. Local requests are still honored.

Notes:

1. Before using this flag, refer to the “Restrictions” section that follows.
2. This option should be used only under direct guidance from IBM Service. It should never be used under normal circumstances.

–M Sets the virtual shared disk maximum IP message size. This is the largest sized block of data the virtual shared disk sends over the network for an I/O request. This limit also affects local virtual shared disk I/O block size. The value is in bytes and must not be greater than the maximum transmission unit (MTU) size of the network. All nodes should use the same value. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/ctlvsd

Examples

1. To display the current parameters, enter:

   ```
   ctlvsd
   ```

   The system displays a message similar to the following:

   ```
   The minimum buddy buffer size is 4096.
   The maximum buddy buffer size is 65536.
   The total buddy buffer size is 4 max buffers, 262144 bytes.
   ```

2. To display the current IP routing table, enter:

   ```
   ctlvsd -t
   ```

   The system displays the following information:

   ```
   Route cache information:

    destination  interface  ref  status  direct/gateway   min managed mbuf
        1           ml0      2     Up        Direct             256
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **lsvsd**, **startrpdomain**, **startrpnode**, **statvsd**

Refer to *RSCT for AIX: Managing Shared Disks* for information about tuning virtual shared disk performance.

defvsd

Purpose

Designates a node as either having or using a virtual shared disk.

Syntax

defvsd *logical_volume_name global_group_name vsd_name*

Description

This command is run to specify logical volumes residing on globally accessible volume groups to be used as virtual shared disks.

You can use the System Management Interface Tool (SMIT) to run the **defvsd** command. To use SMIT, enter:

```
smit vsd_data
```

and select the **Define a Virtual Shared Disk** option.

Parameters

logical_volume_name
Is the name of the logical volume you want to specify as a virtual shared disk. This logical volume must reside on the global volume group indicated. The length of the name must be less than or equal to 15 characters.

global_group_name
Is the name of the globally-accessible volume group previously defined by the **vsdvg** command where you want to specify a virtual shared disk. The length of the name must be less than or equal to 31 characters.

vsd_name
Specifies a unique name for the new virtual shared disk. This name must be unique within the RSCT peer domain, and, in order to avoid possible future naming conflicts, should also be unique across the overall cluster. The suggested naming convention is **vsdnn***gvg_name*. The length of the name must be less than or equal to 31 characters.

Note: If you specify a *vsd_name* that is already the name of another device, the **cfgvsd** command will be unsuccessful for that virtual shared disk. This error ensures that the special device files created for the name do not overlay and destroy files of the same name representing some other device type (such as a logical volume).

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/defvsd

Examples

1. The following example specifies that, on the globally accessible volume group **vg1n1**, the logical volume known as **lv1vg1n1** is used as a virtual shared disk named **vsd1vg1n1**.

   ```
   defvsd lv1vg1n1 vg1n1 vsd1vg1n1
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**, **undefvsd**, **vsdatalst**, **vsdvg**

fencevsd

Purpose

Prevents an application running on a node or group of nodes from accessing a virtual shared disk or group of virtual shared disks.

Syntax

fencevsd {**–a** | **–v** *vsd_name_list*} **–n** *node_list*

Description

Under some circumstances, the system may believe a node has stopped functioning and begin recovery procedures, when the node is actually operational, but cut off from communication with other nodes running the same application. In this case, the problem node must not be allowed to serve requests for the virtual shared disks it normally serves until recovery is complete and the other nodes running the application recognize the problem node as operational. The **fencevsd** command prevents the problem node from filling requests for its virtual shared disks.

This command can be run from any node in the RSCT peer domain where the recoverable virtual shared disk subsystem is running.

Flags

–a Specifies all virtual shared disks.

–v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

–n *node_list* Specifies one or more node numbers, separated by commas.

Security

You must have **root** authority to run this command.

Restrictions

You must issue this command from a node in the peer domain that has an active recoverable virtual shared disk subsystem.

Location

/opt/rsct/vsd/bin/fencevsd

Examples

To fence the virtual shared disks vsd1 and vsd2 from node 5, enter:

```
fencevsd -v vsd1,vsd2 -n 5
```

Related information

Books: *RSCT for AIX: Managing Shared Disks* , for information about the recoverable virtual shared disk subsystem and how you can use the **fencevsd** and **unfencevsd** commands to preserve data integrity during application recovery

Commands: **lsvsd**, **unfencevsd**

ha_vsd

Purpose

Starts and restarts the Recoverable virtual shared disk subsystem. This includes configuring virtual shared disks and activating the recoverability subsystem.

Syntax

ha_vsd [**reset**]

Description

Use this command to start the recoverable virtual shared disk software after you install it, or, with the **reset** option, to stop and restart the program.

Parameters

reset Stops and restarts the recoverable virtual shared disk subsystem.

Exit status

0 Indicates the successful completion of the command.

1 Indicates that an error occurred.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/ha_vsd

Examples

To stop the recoverable virtual shared disk subsystem and restart it, enter:

```
ha_vsd reset
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **ha.vsd**, **startrpdomain**, **startrpnode**

ha.vsd

Purpose

Queries and controls the activity of the rvsd daemon of the recoverable virtual shared disk subsystem.

Syntax

ha.vsd {**adapter_recovery [on** | **off]** | **debug [off]** | **mksrc** | **query** | **quorum** *n* | **qsrc** | **refresh [noquorum]** | **reset** | **reset_quorum** | **rmsrc** | **start** | **stop** | **trace [off]**}

Description

Use this command to display information about the recoverable virtual shared disk subsystem, to change the number of nodes needed for quorum, and to change the status of the subsystem.

Parameters

adapter_recovery [on | off]
Enables or disables communication adapter recovery. The default is **on**.

The recoverable virtual shared disk subsystem must be restarted for this operand to take effect.

debug [off] Specify **debug** to redirect the recoverable virtual shared disk subsystem's standard output and standard error to the console and cause the recoverable virtual shared disk subsystem to not respawn if it exits with an error. (You can use the **lscons** command to determine the current console.)

The recoverable virtual shared disk subsystem must be restarted for this operand to take effect.

Once debugging is turned on and the recoverable virtual shared disk subsystem has been restarted, **ha.vsd trace** should be issued to turn on tracing.

Use this operand under the direction of your IBM service representative.

Note: The default when the node is booted is to have standard output and standard error routed to the console. If debugging is turned off standard output and standard error will be routed to **/dev/null** and all further trace messages will be lost. You can determine if debug has been turned on by issuing **ha.vsd qsrc**. If debug has been turned on the return value will be:

```
action = "2"
```

mksrc Uses **mkssys** to create the recoverable virtual shared disk subsystem.

query Displays the current status of the recoverable virtual shared disk subsystem in detail.

quorum *n* Sets the value of the quorum, which is the total number of nodes

that must join the group before the virtual shared disks will be activated. Usually, quorum is defined as a majority of the nodes that are defined as virtual shared disk nodes in an RSCT peer domain, but this command allows you to override that definition.

The Recoverable virtual shared disk subsystem must be in the active state when you issue this command. This is not a persistent change.

qsrc Displays the System Resource Controller (SRC) configuration of the Recoverable virtual shared disk daemon.

refresh [noquorum]
Uses the **refresh** command to asynchronously start a refresh protocol to all running recoverable virtual shared disk subsystems. The quorum will be reset before the refresh occurs, unless **noquorum** is specified. Use **ha.vsd query** to check for completion. The following items are refreshed in the device driver:
1. Nodes that have been added or deleted
2. Virtual shared disks that have been added or deleted
3. Changed attribute size_in_MB for virtual shared disks

reset Stops and restarts the recoverable virtual shared disk subsystem.

reset_quorum
Resets the default quorum.

rmsrc Uses **rmssys** to remove the recoverable virtual shared disk subsystem.

start Starts the recoverable virtual shared disk subsystem.

stop Stops the recoverable virtual shared disk subsystem.

trace [off] Requests or stops tracing of the recoverable virtual shared disk subsystem. The recoverable virtual shared disk subsystem must be in the active state when this command is issued.

This operand is only meaningful after the **debug** operand has been used to send standard output and standard error to the console and the recoverable virtual shared disk subsystem has been restarted.

Exit status

0 Indicates the successful completion of the command.

nonzero Indicates that an error occurred.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/ha.vsd

Examples

1. To stop the recoverable virtual shared disk subsystem and restart it, enter:

   ```
   ha.vsd reset
   ```

   The system returns the messages:

   ```
   Waiting for the rvsd subsystem to exit.
   rvsd subsystem exited successfully.
   Starting rvsd subsystem.
   rvsd subsystem started PID=xxx.
   ```

2. To change the quorum to five nodes of the RSCT peer domain, enter:

   ```
   ha.vsd quorum 5
   ```

   The system returns the message:

   ```
   Quorum has been changed from 8 to 5.
   ```

3. To query the rvsd subsystem, enter:

   ```
   ha.vsd query
   ```

   The system displays a message similar to the following:

   ```
   Subsystem         Group            PID     Status
         rvsd              rvsd             18320   active
         rvsd(vsd): quorum= 9/4, active=1, state=idle, isolation=member,
                    NoNodes=10, lastProtocol=nodes_failing,
                    adapter_recovery=on, adapter_status=up,
                    RefreshProtocol has never been issued from this node,
                    Running function level 4.2.0.0.
   ```

   where:

   quorum Is the number of total nodes or server nodes that must join the group before virtual shared disks will be activated. In the system output above, `quorum 9/4` indicates the total number of nodes (9) and the number of server nodes (4).

   active Indicates the activation status of the group that is being joined:

   - **0:** the group is not active (quorum has not been met).
   - **1:** the group is active and the shared disks have been activated.

   state Indicates the current protocol that is running.

   isolation Indicates the group membership status

   - **isolated:** a group ″join″ has not been proposed.
   - **proposed:** a group ″join″ has been proposed.
   - **member:** we are a member (provider) of the group.

   NoNodes Indicates the number of nodes that have joined the group

   lastProtocol Indicates the last protocol that was run across the group.

   adapter_recovery
   Indicates communication adapter recovery support:

   - **on:** adapter recovery is enabled.
   - **off:** adapter recovery is disabled.

   adapter_status
   Indicates communication adapter status:

up: the adapter is up.

down: the adapter is down.

unknown: the adapter status is unknown.

RefreshProtocol ...
Indicates whether a refresh protocol has been issued from this node. If so, the date and time of success or error will be displayed.

Running function level
Indicates the function level that the subsystem is running, in version, release, modification, fix level format (vrmf). (Coexistence with lower levels of the subsystem, may restrict us to running at a reduced function level.)

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **ha_vsd**, **startrpdomain**, **startrpnode**

lsvsd

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.

Parameters

vsd_name Specifies a virtual shared disk. This parameter is valid only with the **–l** and **–s** flags.

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.

Security

You must be in the AIX **bin** group to run this command.

Location

/opt/rsct/vsd/bin/lsvsd

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

Related information

Commands: **cfgvsd**, **preparevsd**, **resumevsd**, **startvsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

preparevsd

Purpose

Makes a virtual shared disk available.

Syntax

preparevsd {**–a** | *vsd_name*...}

Description

The **preparevsd** command brings the specified virtual shared disks from the stopped state to the suspended state. The virtual shared disks are made available. Open and close requests are honored, while read and write requests are held until the virtual shared disks are brought to the active state. If they are in the suspended state, this command leaves them in the suspended state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Prepare a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk. If the virtual shared disk is not in the stopped state, you will get an error message.

Flags

–a Specifies that all the virtual shared disks in the stopped state are to be prepared.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/preparevsd

Examples

To bring the virtual shared disk **vsd1vg1n1** from the stopped state to the suspended state, enter:

```
preparevsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **cfgvsd**, **lsvsd**, **resumevsd**, **startrpdomain**, **startrpnode**, **startvsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

removevsd

Purpose

Removes a set of virtual shared disks.

Syntax

removevsd {**–v** *vsd_names* | **–a**} [**–f**]

Description

Use this command to remove the logical volumes associated with the virtual shared disks. Volume groups are not removed with this command.

If the virtual shared disk is configured on any of the nodes on the system partition, this command is unsuccessful, unless the **–f** flag is specified.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Remove a Virtual Shared Disk** option.

Flags

–v Specifies the virtual shared disk name or names that are to be removed by this command.

–a Specifies that the command should remove all virtual shared disks in the RSCT peer domain.

–f Forces the system to unconfigure the virtual shared disks and remove them. If **–f** is not specified and any of the virtual shared disks that are to be removed are configured, the command is unsuccessful.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/removevsd

Examples

To unconfigure and remove all defined virtual shared disks in a system or system partition, enter:

```
removevsd -a -f
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **createvsd**, **startrpdomain**, **startrpnode**

resumevsd

Purpose

Activates an available virtual shared disk.

Syntax

resumevsd [**–p** | **–b** | **–l** *server_list*] {**–a** | *vsd_name* ...}

Description

The **resumevsd** command brings the specified virtual shared disks from the suspended state to the active state. The virtual shared disks remains available. Read and write requests which had been held while the virtual shared disk was in the suspended state are resumed.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Resume a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk.

Flags

–p Specifies that the primary server node defined for the global volume group is to be the active server. The **–p** flag is not valid for CVSD.

–b Specifies that the secondary server node defined for the global volume group is to be the active server. The **–b** flag is not valid for CVSD.

–a Specifies that all the virtual shared disks that have been defined are to be resumed.

–l Passes the **server_list** to the driver.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/resumevsd

Examples

To bring the virtual shared disk **vsd1vg1n1** from the suspended state to the active state, enter:

```
resumevsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **startrpdomain**, **startrpnode**, **startvsd**, **stopvsd**, **suspendvsd**, **ucfgvsd**

rvsdrestrict

Purpose

Displays and sets the run level of the recoverable virtual shared disk subsystem. This command must be issued before the RVSD subsystem will start.

Syntax

rvsdrestrict { **–l** | **–s** { **RVSD4.1** | **RVSD4.2** | **reset** } }

Description

The **rvsdrestrict** command is used to restrict the level at which the recoverable virtual shared disk subsystem will run. If a node has a lower level of the RVSD software installed than what is set with this command, the RVSD subsystem will not start on that node.

This command does not dynamically change RVSD subsystem run levels across the peer domain. An RVSD subsystem instance will only react to this information after being restarted. If your peer domain runs at a given level and you want to override this level, you must:

1. Stop the RVSD subsystem on all nodes.
2. Override the level.
3. Restart the RVSD subsystem.

Flags

–l Lists the current RVSD subsystem run level.

–s Sets the RVSD subsystem run level.

Standard output

Current RVSD subsystem run level.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see the *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/rvsdrestrict

Examples

1. To set the RVSD subsystem run level to **RVSD4.2**, run this command:

```
rvsdrestrict -s RVSD4.2
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**

startvsd

Purpose

Makes a virtual shared disk available and activates it.

Syntax

startvsd [–p | **–b]** {**–a** | *vsd_name* ...}

Description

The **startvsd** command makes the specified virtual shared disks available and activates them. It is equivalent to running the **preparevsd** command followed by the **resumevsd** command on the specified virtual shared disk.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Start a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk.

Flags

–p Specifies that the primary server node defined for the global volume group is to be the active server.

See *RSCT for AIX: Managing Shared Disks* for more information.

–b Specifies that the secondary server node defined for the global volume group is to be the active server.

–a Specifies that all virtual shared disks that have been defined are to be started.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/startvsd

Examples

To make available and activate the virtual shared disk **vsd1vg1n1**, enter:

```
startvsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startrpdomain**, **startrpnode**, **stopvsd**, **suspendvsd**, **ucfgvsd**

statvsd

Purpose

Displays virtual shared disk device driver statistics of a node.

Syntax

statvsd

Description

The **statvsd** command displays virtual shared disk statistics of a node. For example, on a busy server an increasing number of ″requests queued waiting for a buddy buffer″ is normal and does not necessarily imply a problem. Of more value is the ″average buddy buffer wait_queue size″ which is the number of requests queued for a buddy buffer when the **statvsd** command was issued. See the ″Examples″ section for the meaning of output lines.

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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/statvsd

Examples

The following examples display virtual shared disk device driver statistics.

1. The header line indicates the version and release of the code. For example:

   ```
   VSD driver (vsdd):  IP/SMP
   Version:4 Release:1
   ```

2. The level of virtual shared disk parallelism defaults to 9 and is the buf_cnt parameter on the uphysio call that the device driver makes in the kernel. For example:

   ```
   9 vsd parallelism
   ```

3. The maximum IP message size in bytes. For example:

   ```
   61440 vsd max IP message size
   ```

4. The number of requests that had to wait for a request block. For example:

   ```
   61440 vsd max IP message size
   ```

5. The number of requests that had to wait for a pbuf (a buffer used for the actual physical I/O request submitted to the disk). For example:

   ```
   0 requests queued waiting for a pbuf
   ```

6. The number of requests that had to wait for a buddy buffer. A buffer that is used on a server to temporarily store date for I/O operations originating at a client node. For example:

   ```
   2689 requests queued waiting for a buddy buffer
   ```

7. The number of requests queued for a buddy buffer when the **statvsd** command was issued. For example:

   ```
   0 average buddy buffer wait_queue size
   ```

8. The number of requests that a server has rejected, typically because of an out-of-range sequence number or an internal problem. For example:

   ```
   4 rejected requests
   ```

9. The number of responses that a client has rejected. Typically because a response arrived after a retry was already sent to the server. For example:

   ```
   0 rejected responses
   ```

10. The number of requests that were placed on the rework queue. For example:

    ```
    0 requests rework
    ```

11. The number of read requests that were not on a 64-byte boundary. For example:

    ```
    0 64 byte unaligned reads
    ```

12. The number of requests that got a DMA shortage. This condition would require the I/O operation to be executed in nonzero copy mode. For example:

    ```
    0 DMA space shortage
    ```

13. The number of requests that have timed out. The current timeout period is approximately 15 minutes. For example:

    ```
    0 timeouts
    ```

14. There are a fixed number of retries. The retries counters display the number of requests that have been retried for that particular ″retry bucket.″ Numbers appearing further to the right represent requests that have required more retries. When a request exhausts its number of retries, it gets recorded as a timeout. For example:

    ```
    retries:  0 0 0 0 0 0 0 0 0

              0 total retries
    ```

15. Sequence numbers are internally used by the device driver. These numbers are managed by the device driver and the recoverable virtual shared disk subsystem. For example:

    ```
    Non-zero Sequence Numbers

    node#     expected     outgoing     outcase?     Incarnation:0
     11        125092         0            |

     11 Nodes Up with zero sequence numbers: 1 3 5 7 9 11 12 13 14 15 16
    ```

Related information

Books:

- *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain
- *RSCT for AIX: Managing Shared Disks*, for information about tuning virtual shared disk performance

Commands: **ctlvsd**, **startrpdomain**, **startrpnode**, **vsdnode**

stopvsd

Purpose

Makes a virtual shared disk unavailable.

Syntax

stopvsd {**–a** | *vsd_name* ...}

Description

The **stopvsd** command brings the specified virtual shared disks from the suspended state to the stopped state. This makes the virtual shared disks unavailable. All applications that have outstanding requests for the virtual shared disk see these requests terminate with error. Read and write requests return errors with **errno** set to **ENODEV**. If the virtual shared disk is in the stopped state, this command leaves it in the stopped state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Stop a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk. If the virtual shared disk is not is the suspended state, you get an error message.

Flags

–a Specifies that all virtual shared disks in the suspended state are to be stopped.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/stopvsd

Examples

To bring the virtual shared disk **vsd1vg1n1** from the suspended state to the stopped state, enter:

```
stopvsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startrpdomain**, **startrpnode**, **startvsd**, **suspendvsd**, **ucfgvsd**

suspendvsd

Purpose

Deactivates an available virtual shared disk.

Syntax

suspendvsd {**–a** | *vsd_name*...}

Description

The **suspendvsd** command brings the specified virtual shared disks from the active state to the suspended state. They remain available. Read and write requests which were active while the virtual shared disk was active are suspended and held. Subsequent read and write operations are also held. If the virtual shared disk is in the suspended state, this command leaves it in the suspended state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Suspend a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk. If the virtual shared disk is not in the active state, you get an error message.

Flags

–a Specifies that all the virtual shared disks in the active state are to be suspended.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/suspendvsd

Examples

To bring the virtual shared disk **vsd1vg1n1** from the active state to the suspended state, enter:

```
suspendvsd vsd1vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startrpdomain**, **startrpnode**, **startvsd**, **stopvsd**, **ucfgvsd**

ucfgvsd

Purpose

Unconfigures a virtual shared disk.

Syntax

ucfgvsd {**–a** | *vsd_name* ...}

Description

The **ucfgvsd** command unconfigures the specified virtual shared disks. The specified virtual shared disks must be in the stopped state to be unconfigured. This command does not change any virtual shared disk definitions. It moves virtual shared disks from the stopped state to the defined state.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Unconfigure a Virtual Shared Disk** option.

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Parameters

vsd_name Specifies a virtual shared disk. The disk specified must be in the stopped state. If all disks have been unconfigured, and you specify VSD0, this command will attempt to unload the device driver from the kernel.

Flags

–a Specifies that all virtual shared disks in the stopped state are to be unconfigured.

Security

You must have **root** authority to run this command.

Restrictions

Under normal circumstances, you should not issue this command. The recoverable virtual shared disk subsystem uses this command to manage shared disks in a controlled manner. If you issue this command, the results may be unpredictable.

Location

/opt/rsct/vsd/bin/ucfgvsd

Examples

To unconfigure the virtual shared disk **vsd1vg1n1** in the stopped state, enter:

```
ucfgvsd vsd1vg1n1
```

Related information

Commands: **cfgvsd**, **lsvsd**, **preparevsd**, **resumevsd**, **startvsd**, **stopvsd**, **suspendvsd**

undefvsd

Purpose

Undefines a virtual shared disk.

Syntax

undefvsd *vsd_name* ...

Description

This command is used to remove a virtual shared disk definition and any special device files from **/dev** for the given *vsd_names* on all the virtual shared disk nodes. The virtual shared disks must be unconfigured and in the defined state on all the virtual shared disk nodes.

You can use the System Management Interface Tool (SMIT) to run the **undefvsd** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Undefine a Virtual Shared Disk** option.

Parameters

vsd_name Specifies the virtual shared disk whose underlying logical volume you no longer want to be globally accessed by any virtual shared disk nodes.

Security

You must have **root** authority to run this command.

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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/usr/lpp/vsd/bin/undefvsd

Examples

To delete the information associated with the virtual shared disk **vsd1vg2n1**, enter:

```
undefvsd vsd1vg2n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **defvsd**, **startrpdomain**, **startrpnode**

unfencevsd

Purpose

Gives applications running on a node or group of nodes access to a virtual shared disk or group of virtual shared disks that were previously fenced from applications running on those nodes.

Syntax

unfencevsd {**–a** | **–v** *vsd_name_list*} {**–n** *node_list* [**–f**] }

Description

Under some circumstances, the system may believe a node has become inoperable and may begin recovery procedures when the node is actually operational, but is cut off from communication with other nodes running the same application. In this case, the problem node must not be allowed to serve requests for the virtual shared disks it normally manages until recovery is complete and the other nodes running the application recognize the problem node as operational. The **fencevsd** command prevents the problem node from filling requests for its virtual shared disks. The **unfencevsd** command allows fenced nodes to regain access to the virtual shared disks.

You can issue this command from any node that is online in the peer domain.

Flags

–a Specifies all virtual shared disks.

–f Allows a fenced node to unfence itself.

–n *node_list* Specifies one or more node numbers separated by commas.

–v *vsd_name_list*
Specifies one or more virtual shared disk names, separated by commas.

Security

You must have **root** authority to run this command.

Location

/opt/rsct/vsd/bin/unfencevsd

Examples

1. To unfence node 5 from the virtual shared disks vsd1 and vsd2, enter:
   ```
   unfencevsd -v vsd1,vsd2 -n 5
   ```
2. To unfence node 7 from the virtual shared disks vsd1 and vsd2 when the **unfencevsd** command must be entered from node 7, enter:
   ```
   unfencevsd -v vsd1,vsd2 -n 7 -f
   ```

Related information

Commands: **fencevsd**, **lsvsd**

Refer to *RSCT for AIX: Managing Shared Disks* for information about the recoverable virtual shared disk subsystem and how you can use the **fencevsd**,

unfencevsd, and **lsfensevsd** commands to preserve data integrity during application recovery.

updatevsdnode

Purpose

Modifies virtual shared disk subsystem options.

Syntax

updatevsdnode –n { **ALL** | *node_number*[**,***node_number*,...] } { [**–a** *adapter_name*] [**–b** *min_buddy_buffer_size*] [**–x** *max_buddy_buffer_size*] [**–s** *max_buddy_buffers*] [**–M** *max_IP_msg_size*] [**–r** *max_read_count*] [**–w** *max_write_count*] [**–p** *max_rdrsp_count*] } [**–f**] [**–c** { *cluster_name* | **NONE** }]

Description

Use **updatevsdnode** to modify virtual shared disk subsystem options.

Note: This command modifies the subsystem options only. To configure the virtual shared disks effectively, you must first unconfigure all the virtual shared disks, unload the device driver, and then reconfigure the shared disks.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the **Set/Show Virtual Shared Disk Device Driver Operational Parameters** option or the **Update virtual shared disk Device Driver Node Parameters** option.

Flags

–n { ALL | *node_number***[,***node_number***,...] }**
: Specifies one or more nodes for which you want to update information. You can use the **lsrpnode -i** command to find out the node number of the node you are running on. Use **-n ALL** to update information on all of the nodes in the RSCT peer domain.

–a *adapter_name*
: Specifies the adapter name to be used for virtual shared disk communication with the node or nodes. You must specify **ml0** as the adapter name.

–b *min_buddy_buffer_size*
: Specifies the smallest buddy buffer size that a server will use to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. IBM suggests using the value of 4096 (4KB).

–x *max_buddy_buffer_size*
: Specifies the largest buddy buffer size that a server will use to satisfy a remote request. This value must be a power of 2 and greater than or equal to *min_buddy_buffer_size*. The recommended value is 262144 (256KB). This value must be the same on all nodes in the RSCT peer domain.

–s *max_buddy_buffers*
: Specifies the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim

additional space as needed. The suggested starting value for a 32-bit kernel is 128 256KB buffers. The suggested value is 2000 256KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

The **statvsd** command indicates whether remote requests are queueing while waiting for buddy buffers.

–M *max_IP_msg_size*
Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

–r *max_read_count*
Specifies the maximum number of outstanding concurrent read operations that a client node will allow to be active. Additional read operations will be queued. By limiting the amount of data that a client node receives, this provides a mechanism for preventing resource overruns.

–w *max_write_count*
Specifies the maximum number of outstanding concurrent write operations that a server node will allow to be active. Additional write operations will be queued. By limiting the amount of data that a server node receives, this provides a mechanism for preventing resource overruns.

–p *max_rdrsp_count*
Specifies the maximum number of outstanding concurrent read responses that a server node will allow to be active. Additional read responses will be queued. By limiting the amount of data that a server node sends, this provides a mechanism for preventing resource overruns.

–f
Forces updates to virtual shared disk subsystem options by reconfiguring one or more virtual shared disks on all nodes in the RSCT peer domain on which virtual shared disks are currently configured.

–c { *cluster_name* | NONE }
Changes the cluster to which the node belongs. A cluster name is required for Serial Storage Architecture (SSA) disks only. Use **-c NONE** to remove the node from the cluster.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/lpp/vsd/bin/updatevsdnode

Examples

1. To increase the buddy buffer size to 48 maximum-sized buddy buffers on node 3, enter:

   ```
   updatevsdnode -n 3 -s 48
   ```

   The device driver must be unconfigured from the kernel and reloaded to have this change go into effect.
2. To change the maximum number of read operations to **10** for all nodes in the domain, enter:

   ```
   updatevsdnode -n ALL -r 10
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **lsrpnode**, **lsvsd**, **startrpdomain**, **startrpnode**, **statvsd**, **vsdatalst**, **vsdnode**

updatevsdtab

Purpose

Changes the virtual shared disk subsystem attributes.

Syntax

updatevsdtab {**–v** *vsd_names* | **–a**} {[**–s**]} [**–f**]

Description

Use this command to update the virtual shared disk size. When you change the virtual shared disk size using the **updatevsdtab** command, the change will not take effect until the virtual shared disk is unconfigured and configured again.

If the **–f** flag is specified, the virtual shared disks involved will be reconfigured on all nodes that are up and initially had these virtual shared disks configured.

You can use the System Management Interface Tool (SMIT) to run this command. To use SMIT, enter:

```
smit vsd_mgmt
```

and select the Set/Show virtual shared disk Device Driver Operational Parameters option or the Update virtual shared disk Options option.

Flags

–v *vsd_names*	Specifies a list of virtual shared disk names to be updated.
–a	Specifies that the option is to be changed on all nodes of the system or system partition.
–s	Updates the virtual shared disk size after the associated logical volume size is changed.
–f	Forces changes by reconfiguring a virtual shared disk on all nodes in the current system partition on which the virtual shared disk is configured.

Security

You must have **root** authority to run this command.

Location

/usr/lpp/csd/bin/updatevsdtab

Examples

1. To reset the size of the virtual shared disk named USER1n3, enter:

   ```
   updatevsdtab -v USER1n3 -s
   ```

Related information

Commands: **defvsd**, **updatevsdnode**

updatevsdvg

Purpose

Changes virtual shared disk global volume group characteristics.

Syntax

updatevsdvg { **–a** | **–g** *global_volgrp* { **–k VSD –p** *primary_node* **–b** *secondary_node* | **–k CVSD –l** *server_list* [**–c** *cluster_name*] } }

Description

The **updatevsdvg** command changes virtual shared disk global volume group characteristics. This command allows you to change global volume groups from concurrent virtual shared disk volume groups to serial-access (or nonconcurrent) virtual shared disk volume groups, and the other way around. This command can be used whenever server node numbers change, such as replacing or re-cabling servers where the new server numbers are different, or when you need to delete a server.

This command performs the following operations:

1. Suspends all virtual shared disks that are part of this volume group
2. Stops all virtual shared disks that are part of this volume group
3. Issues the **varyoffvg** command for the volume group
4. Verifies that the volume group exists on the new servers and tries to import the volume group if it does not exist
5. Updates the global volume group characteristics
6. Issues the **varyonvg** command for the volume group to the appropriate servers
7. Starts all virtual shared disks that are part of this volume group

Notes:

1. If you issue this command with the **-a** flag, the recoverable virtual shared disk subsystem should not be active. Otherwise, this command can be run while the recoverable virtual shared disk subsystem is active, as long as no application is using the virtual shared disks that are part of the volume group being updated.
2. Concurrent virtual shared disks are supported for disks that have implemented the SCSI-3 persistent reserve model of the AIX SCSI device drivers, and for SSA (Serial Storage Architecture) disks.

Flags

–a

Specifies that persistent reserve information should be reestablished in the object data manager (ODM) for all VSD volume groups served by this node. This flag is intended for the initial setup phase of allowing multiple clusters to access the same virtual shared disks. It is also useful for recovery after the device ODM entries have been removed inadvertently.

This flag causes all of the volume groups served by the node to be varied offline. The volume groups will be varied offline on this node and on all other servers for the volume groups. For this reason, you

	should stop the recoverable virtual shared disk subsystem before issuing the **updatevsdvg** command with this flag.
–b *secondary_node*	Specifies the secondary node.
–c *cluster_name*	Specifies the cluster name for the server nodes that will be serving concurrently accessed shared disks. This flag is applicable only for SSA (Serial Storage Architecture) disks, and a *cluster_name* must be specified for SSA.
–g *global_volgrp*	Specifies an existing global volume group name.
–k VSD | CVSD	Specifies whether the volume group will be of type concurrent virtual shared disk or serial-access (nonconcurrent) virtual shared disk.
–l *server_list*	Specifies a colon-separated list of servers for concurrent virtual shared disks.
–p *primary_node*	Specifies the primary node.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/updatevsdvg

Examples

1. To change the global volume group named **ess_gvg** from a virtual shared disk global volume group to a concurrent global volume group with three servers, assuming that the disks are cabled correctly and that the disk subsystem supports persistent peserve such as ESS disks, enter:

   ```
   updatevsdvg -g ess_gvg -k CVSD -l 9:17:21
   ```

2. To remove a server from an SSA global volume group named **ssa_gvg**, where the original server list is **9:10** and belongs to an SSA cluster named **cluster9_10**, (that is, the command **vsdatalst -c** shows SSA cluster information), enter:

   ```
   updatevsdvg -g ssa_gvg -k CVSD -l 9 -c cluster9_10
   ```

3. To change a concurrent global volume group named **ess_gvg** back to a virtual shared disk global volume group, where the original server list is **9:17:21**, the new primary node number is 9, and the new secondary node number is 21, enter:

   ```
   updatevsdvg -g ess_gvg -k VSD -p 9 -b 21
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**, **vsdatalst**, **vsdvg**

vsdatalst

Purpose

Displays virtual shared disk subsystem information.

Syntax

vsdatalst {**–g** | **–n** | **–v** | **–c**}

Description

Use this command to display one of several kinds of information to standard output.

You can use the System Management Interface Tool (SMIT) to run the **vsdatalst** command. To use SMIT, enter:

```
smit list_vsd
```

and select the option for the kind of virtual shared disk SDR information you want to see.

Flags

Only one of the following flags can be specified with each invocation of **vsdatalst**:

–g Displays the following global volume group data:

global_group_name,
local_group_name,
primary_server_node,
secondary_server_node. (This is only enabled with the recoverable virtual shared disk subsystem.)
eio_recovery
recovery
CVSD *server_list*

–n Displays the following Node data:

node_number,
host_name,
adapter_name,
min_buddy_buffer_size,
max_buddy_buffer_size,
max_buddy_buffers.

–v Displays the following definition data:

vsd_name,
logical_volume_name,
global_group_name,
minor_number.

–c Displays the following cluster information:

node_number
cluster_name

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdatalst

Examples

1. To display global volume group date, enter:

   ```
   vsdatalst -g
   ```

   The system displays a message similar to the following:

```
VSD Global Volume Group Information
                                          Server Node Numbers
Global Volume Group name   Local VG name   primary   backup eio_recovery  recovery  server_list  vsd_type
------------------------    ------------   -------   ------ ------------  --------  -----------  --------
gpfs0gvg                     gpfs0vg         1         2          0          0        0            VSD
gpfs1gvg                     gpfs1vg         2         1          0          0        0            VSD
gpfs3gvg                     gpfs3vg         1         0          0          0        1:2          CVSD
```

2. To display global volume group date, enter:

   ```
   vsdatalst -n
   ```

   The system displays a message similar to the following:

```
VSD Node Information
                                                    Buddy Buffer
  node                 VSD      IP packet minimum maximum # maxbufs
number host_name       adapter    size     size    size
------ --------------- -------- --------- ------- ------- ---------
     1 host1            ml0       61440    4096   262144    128
     2 host2            ml0       61440    4096   262144    128
```

3. To display global volume group date, enter:

   ```
   vsdatalst -v
   ```

   The system displays a message similar to the following:

```
VSD Table
VSD name                      logical volume  Global Volume Group             minor# size_in_MB
----------------------------- --------------- ------------------------------- ------ ----------
gpfs0vsd                      gpfs0lv         gpfs0gvg                             3 4096
gpfs1vsd                      gpfs1lv         gpfs1gvg                             1 4096
gpfs3vsd                      gpfs3lv         gpfs3gvg                             4 4096
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **lsvsd**, **startrpdomain**, **startrpnode**, **updatevsdnode**, **vsdnode**

vsdchgserver

Purpose

Switches the server function for one or more virtual shared disks from the node that is currently acting as the server node to the other.

Syntax

vsdchgserver **–g** *vsd_global_volume_group_name* **–p** *primary_node*

[**–b** *secondary_node*] [**–o** *EIO_recovery*]

Description

The **vsdchgserver** command allows the serving function for a global volume group defined on a primary node to be taken over by the secondary node, or to be taken over by the primary node from the secondary node. This allows an application to continue to use virtual shared disks in situations where the cable or adapter between the physical disks and one of the attached nodes is not working.

The recoverable virtual shared disk subsystem automatically updates the virtual shared disk devices if, and only if, the **vsdchgserver** command is used to flip the currently-defined primary node and secondary node in the global volume group specified in the **–g** flag.

Flags

–g Specifies the Global Volume Group name for the volume group that represents all the virtual shared disks defined on a particular node.

–p Specifies the primary server node number for the global volume group.

–b Specifies the secondary node number for the global volume group. If the **-b** flag is not specified, the secondary node definition will be removed.

–o Specified as **0**, for no recovery on an EIO error, or **1**, for recovery on an EIO error.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdchgserver

Examples

To change the primary server node for the global volume group node12vg to node 1 and the secondary node to node 2, with EIO recovery, enter:

```
vsdchgserver -g node12vg -p 1 -b 2 -o 1
```

Related information

Books:

- *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain
- *RSCT for AIX: Managing Shared Disks*, for information about how to use this command in writing applications

Commands: **startrpdomain**, **startrpnode**

vsdelnode

Purpose

Removes virtual shared disk information for a node or series of nodes.

Syntax

vsdelnode *node_number* ...

Description

This command is used to remove virtual shared disk data for a node or series of nodes.

The **vsdelnode** command makes the listed nodes no longer virtual shared disk nodes so that no virtual shared disks can be accessed from them. This command is unsuccessful for any nodes that are servers for any global volume groups.

You can use the System Management Interface Tool (SMIT) to run the **vsdelnode** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Delete Virtual Shared Disk Node** Information option.

Parameters

node_number Specifies the node number of the node whose virtual shared disk information you want to remove.

Security

You must have **root** authority to run this command.

Restrictions

The recoverable virtual shared disk subsystem must be stopped on the node or nodes you are deleting. Otherwise, the results may be unpredictable. For more information, see *RSCT for AIX: Managing Shared Disks*.

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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdelnode

Examples

To delete virtual shared disk node information for nodes **3** and **6**, enter:

```
vsdelnode 3 6
```

Related information

Books:

- *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain
- *RSCT for AIX: Managing Shared Disks*

Commands: **startrpdomain**, **startrpnode**, **vsdatalst**, **vsdnode**

vsdelvg

Purpose

Removes virtual shared disk global volume group information.

Syntax

vsdelvg [**–f**] *global_group_name* ...

Description

Use this command to remove virtual shared disk global volume group information. If any virtual shared disks are defined on a global volume group, the **vsdelvg** command is unsuccessful unless **–f** is specified. If **–f** is specified, any such virtual shared disks must be unconfigured and in the defined state on all the virtual shared disk nodes to be deleted.

You can use the System Management Interface Tool (SMIT) to run the **vsdelvg** command. To use SMIT, enter:

```
smit delete_vsd
```

and select the **Delete Virtual Shared Disk Global Volume Group Information** option.

Parameters

global_group_name
: Specifies the volume group that you no longer want to be global to the system.

Flags

–f
: Forces the removal of any virtual shared disks defined on this global volume group.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdelvg

Examples

To remove the global volume group **vg1n1**, enter:

```
vsdelvg vg1n1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**, **undefvsd**, **vsdatalst**, **vsdvg**

vsdnode

Purpose

Defines virtual shared disk information for a node or series of nodes.

Syntax

vsdnode *node_number ... adapter_name min_buddy_buffer_size max_buddy_buffer_size max_buddy_buffers max_IP_msg_size* [{ *cluster_name* | **NONE** } *max_read_count max_write_count max_rdrsp_count*]

Description

Use this command to make the specified nodes virtual shared disk nodes and to assign their virtual shared disk operational parameters. If this information is the same for all nodes, run this command once. If the information is different for the nodes, run this command once for each block of nodes that should have the same virtual shared disk information.

You can use the System Management Interface Tool (SMIT) to run the **vsdnode** command. To use SMIT, enter:

```
smit vsd_data
```

and select the **virtual shared disk Node Information** option.

Parameters

node_number Specifies one or more nodes for which you want to set virtual shared disk information. The value you specify for *node_number* must match a valid RSCT remote peer domain node number.

You can use the **lsrpnode -i** command to find out the node number of the node you are running on.

adapter_name Specifies the adapter name to be used for virtual shared disk communication for the nodes. The adapter name must already be defined to the nodes. The nodes involved in virtual shared disk support must be fully connected so that proper communication can take place. Use **ml0** to specify that the virtual shared disk device driver transmits data requests over the SP™ Switch2. The **ml0** adapter will be used the next time the virtual shared disk device driver is loaded.

min_buddy_buffer_size
Specifies the smallest buddy buffer size that a server will use to satisfy a remote request to a virtual shared disk. This value must be a power of 2 and greater than or equal to 4096. IBM suggests using a value of 4096 (4KB). For a 512 byte request, 4KB is excessive. However, recall that a buddy buffer is only used for the short period of time while a remote request is being processed at the server node.

max_buddy_buffer_size
Specifies the largest buddy buffer size that a server will use to satisfy a remote noncached request. This value must be a power of 2 and greater than or equal to the *min_buddy_buffer_size.* IBM suggests using a value of 262144 (256KB). This value depends on

the I/O request size of applications using the virtual shared disks and the network used by the virtual shared disk software.

max_buddy_buffers
Specifies the number of *max_buddy_buffer_size* buffers to allocate. The virtual shared disk device driver will have an initial size when first loaded, and then will dynamically allocate and reclaim additional space as needed. The suggested value is 2000 256KB buffers.

Buddy buffers are only used on the servers. On client nodes you may want to set *max_buddy_buffers* to 1.

Note: The **statvsd** command indicates whether remote requests are queueing while waiting for buddy buffers.

max_IP_msg_size
Specifies the maximum message size in bytes for virtual shared disks. This value must not be greater than the maximum transmission unit (MTU) size of the network. The recommended values are:

- 61440 (60KB) for a switch
- 8192 (8KB) for jumbo frame Ethernet
- 1024 (1KB) for 1500-byte MTU Ethernet

cluster_name **| NONE**
Specifies a cluster name. You must specify a cluster name for server nodes that will be serving concurrently-accessed shared disks. A node can only belong to one cluster. For example, when you have a concurrent access environment, the two servers for the CVSD must both specify the same cluster name. A cluster name is required for Serial Storage Architecture (SSA) disks only.

Specify **NONE** to remove the node from the cluster.

max_read_count
Specifies the maximum number of outstanding concurrent read operations that a client node will allow to be active. Additional read operations will be queued. By limiting the amount of data that a client node receives, this provides a mechanism for preventing resource overruns. In a High Performance Switch (HPS) environment, it is recommended that you set *max_read_count* to a value for which the total amount of concurrent data is less than the RPOOL size. For most systems, the default value does not need to be changed.

max_write_count
Specifies the maximum number of outstanding concurrent write operations that a server node will allow to be active. Additional write operations will be queued. By limiting the amount of data that a server node receives, this provides a mechanism for preventing resource overruns. In an HPS environment, it is recommended that you set *max_write_count* to a value for which the total amount of concurrent data equals the RPOOL size. For most systems, the default value does not need to be changed.

max_rdrsp_count
Specifies the maximum number of outstanding concurrent read responses that a server node will allow to be active. Additional read

responses will be queued. By limiting the amount of data that a server node sends, this provides a mechanism for preventing resource overruns. In an HPS environment, it is recommended that you set *max_read_count* to a value for which the total amount of concurrent data is less than the RPOOL size. For most systems, the default value does not need to be changed.

Security

You must have **root** authority to run this command.

Restrictions

The node specified on this command must already belong to a peer domain, and 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdnode

Examples

1. To define information for a switch network and nodes 1 through 8, enter:
   ```
   vsdnode 1 2 3 4 5 6 7 8 ml0 4096 262144 128 61440
   ```
2. To change the maximum number of read operations to **10**, enter:
   ```
   vsdnode 1 css0 4095 262144 4 61440 NONE 10
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **lsrpnode**, **startrpdomain**, **startrpnode**, **updatevsdnode**, **vsdatalst**, **vsdelnode**

vsdsklst

Purpose

Produces output that shows you the disk resources used by the virtual shared disk subsystem across a peer domain.

Syntax

vsdsklst [**–v**] [**–d**] {**–a** | **–n** *node_number*[, node_number2, ...]}

Description

Use this command to check disk utilization across a peer domain.

Flags

–v Displays only disk utilization information about volume groups and the virtual shared disks associated with them.

–d Displays only disk utilization information about volume groups and the physical disks associated with them.

–a Displays specified information for all nodes in the system or system partition.

–n *node_number*
Lists one or more node numbers for which information is to be displayed.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdsklst

Examples

This command:

```
vsdsklst -dv -a
```

displays the following information on a system that has volume groups and virtual shared disks defined on nodes 1 and 2.

```
c164n12.ppd.pok.ibm.com: Node Number:2; Node Name:c164n12.ppd.pok.ibm.com
c164n12.ppd.pok.ibm.com: Volume group:rootvg; Partition Size:32; Total:271; Free:168
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk0; Total:271; Free:168
c164n12.ppd.pok.ibm.com:     Volume group:testvg is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk5;
c164n12.ppd.pok.ibm.com: Volume group:test1vg; Partition Size:4; Total:537; Free:534
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk2; Total:537; Free:534
c164n12.ppd.pok.ibm.com:         VSD Name:vsd1n2[testnewlv21n2]; Size:1
```

```
c164n12.ppd.pok.ibm.com:         VSD Name:vsd2n2[testlv1n2]; Size:346112.25
c164n12.ppd.pok.ibm.com:         VSD Name:vsd3n2[testlv2n2]; Size:346112.25
c164n12.ppd.pok.ibm.com:     Volume group:vg1 is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk9;
c164n12.ppd.pok.ibm.com:     Volume group:sharkvg is not varied on.
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk7;
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk10;
c164n12.ppd.pok.ibm.com: Volume group:bdhclvg; Partition Size:32; Total:134; Free:102
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk13; Total:134; Free:102
c164n12.ppd.pok.ibm.com: Volume group:gpfs0vg; Partition Size:8; Total:536; Free:0
c164n12.ppd.pok.ibm.com:         Physical Disk:hdisk12; Total:536; Free:0
c164n12.ppd.pok.ibm.com:         VSD Name:gpfs0vsd[gpfs0lv]; Size:352256.75
c164n12.ppd.pok.ibm.com:     Not allocated physical disks:
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk1
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk3
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk4
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk6
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk11
c164n12.ppd.pok.ibm.com:         Physical disk:hdisk15
c164n11.ppd.pok.ibm.com: Node Number:1; Node Name:c164n11.ppd.pok.ibm.com
c164n11.ppd.pok.ibm.com: Volume group:rootvg; Partition Size:32; Total:271; Free:172
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk0; Total:271; Free:172
c164n11.ppd.pok.ibm.com: Volume group:bdhclvg; Partition Size:32; Total:134; Free:102
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk9; Total:134; Free:102
c164n11.ppd.pok.ibm.com:         VSD Name:bdhcvsd1n1[lvbdhcvsd1n1]; Size:45056
c164n11.ppd.pok.ibm.com: Volume group:testvg; Partition Size:16; Total:134; Free:70
c164n11.ppd.pok.ibm.com:         Physical Disk:hdisk13; Total:134; Free:70
c164n11.ppd.pok.ibm.com:     Not allocated physical disks:
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk1
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk2
c164n11.ppd.pok.ibm.com:         Physical disk:hdisk3
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**, **vsdatalst**

vsdvg

Purpose

Defines a virtual shared disk global volume group.

Syntax

vsdvg [**–g** *global_volume_group*] {**–l** *server_list local_group_name* |
local_group_name primary_node [*secondary node* [*eio_recovery*]]}

Description

Use this command to define volume groups for use by the virtual shared disk subsystem. This is done by specifying the local volume group name, the node on which it resides, and the name by which the volume group will be known throughout the cluster.

You can use the System Management Interface Tool (SMIT) to run the **vsdvg** command. To use SMIT, enter:

```
smit vsd_data
```

and select the **Virtual Shared Disk Global Volume Group Information** option.

Parameters

local_group_name
: Specifies the name of a volume group that you want to indicate as being used for virtual shared disks. This name is local to the host upon which it resides. The length of the name must be less than or equal to 15 characters.

primary_node
: Specifies the primary server node number on which the volume group resides. The length of the name must be less than or equal to 31 characters.

secondary_node
: Specifies the secondary server node number on which the volume group resides. The length of the name must be less than or equal to 31 characters.

eio_recovery
: Specifies how the recoverable virtual shared disk subsystem will respond to EIO errors. If *eio_recovery* is set to the value 1 (the default), an EIO error will cause the recoverable virtual shared disk system to flip the current primary node and the secondary node and perform one more retry on the new primary node.

Flags

–g *global_volume_group*
: Specifies a unique name for the new global volume group. This name must be unique across the system partition. It should be unique across the SP, to avoid any naming conflicts during future system partitioning operations. The suggested naming convention is **vg***xx***n***yy*, where *yy* is the node number, and *xx* uniquely numbers the volume groups on that node. If this is not specified, the local

group name is used for the global name. The length of the name must be less than or equal to 31 characters.

–l *server_list* Define the list of servers for CVSD. More than one server indicates the *global_volume_group* is a concurrent volume group.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/opt/rsct/vsd/bin/vsdvg

Examples

1. The following example defines **gpfs1gvg** as a virtual shared disk global volume group with the local volume group **gpfs1vg** accessed from **node1** as the *primary_node* and **node2** as the *secondary_node* .

   ```
   vsdvg -g gpfs1gvg gpfs1vg 1 2
   ```

2. The following example defines **gpfs3gvg** as a virtual shared disk global volume group with the local volume group **gpfs3vg** concurrently accessed from **node1** and **node2**.

   ```
   vsdvg -g gpfs3gvg -l 1:2 gpfs3vg
   ```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**, **updatevsdvg**, **vsdelvg**

vsdvgts

Purpose

Updates the timestamp used by the recoverable virtual shared disk subsystem by reading the timestamp from the volume group descriptor area (VGDA) of the physical disks.

Syntax

vsdvgts [**–a**] [*volgrp*]

Description

Use this command to update the timestamp that the recoverable virtual shared disk subsystem uses to determine if a twin-tailed volume group has changed. When the subsystem detects a change, the recovery scripts export the volume group and then import the volume group.

This command can be used to avoid exporting the volume group and then importing the volume group during recovery in situations where the export and import operations are not really necessary. This command should be used very carefully.

Parameters

volgrp Specifies a volume group. If this operand is not specified, the timestamps for all the volume groups on this node are updated.

Flags

–a Specifies that the timestamps for this volume group for both primary and secondary nodes should be updated. If this flag is not specified, the timestamp is updated on the local node only.

Exit status

0 Indicates the successful completion of the command.

1 Indicates that the program was unable to read one or more timestamps.

Security

You must have **root** authority 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 information about creating and administering an RSCT peer domain, see *RSCT: Administration Guide*.

Location

/usr/lpp/vsd/bin/vsdvgts

Examples

To update the timestamp associated with the virtual shared disk volume group vsdvg1 for just this node, enter:

```
vsdvgts vsdvg1
```

Related information

Books: *RSCT: Administration Guide*, for information about creating and administering an RSCT peer domain

Commands: **startrpdomain**, **startrpnode**

Part 10. Problem determination

Chapter 20. Commands for data collection 543
ctsnap 544

Chapter 21. First failure data capture (FFDC) commands 547
fccheck 548
fcclear 550
fcdecode 554
fcdispfid 556
fcfilter 558
fcinit 560
fclogerr 565
fcpushstk 574
fcreport 582
fcstkrpt 585
fcteststk 588

Chapter 22. FFDC files 591
ct_ffdc.h 592

© Copyright IBM Corp. 2002, 2007

Chapter 20. Commands for data collection

© Copyright IBM Corp. 2002, 2007

ctsnap

Purpose

Gathers configuration, log, and trace information about the Reliable Scalable Cluster Technology (RSCT) components.

Syntax

ctsnap [**-d** *output_directory*] [**-x runrpttr**] [**-h**]

Description

The **ctsnap** command gathers configuration, log, and trace information about the RSCT components that are installed with the base operating system. The **ctsnap** command collects data only on the local node on which it (**ctsnap**) is running. Depending on the programs that are installed, the following components may be included:

- Audit log resource manager (**IBM.AuditRM**)
- Cluster security services (**ctsec**)
- Common information model resource manager (**IBM.CIMRM**)
- Configuration resource manager (**IBM.ConfigRM**)
- Domain management server resource manager (**IBM.DMSRM**)
- Event management (**ha_em**)
- Event response resource manager (**IBM.ERRM**)
- File system resource manager (**IBM.FSRM**)
- First failure data capture (**ct_ffdc**)
- Group services (**cthags**)
- Host resource manager (**IBM.HostRM**)
- Least-privilege resource manager (**IBM.LPRM**)
- Low-level application programming interface (**lapi**)
- Resource monitoring and control (**ctrmc**)
- Sensor resource manager (**IBM.SensorRM**)
- Service Availability Forum (SAF) resource manager (**IBM.SAFRM**)
- Storage resource manager (**IBM.StorageRM**)
- Topology services (**cthats**)
- Virtual shared disk (**vsd**)
- Recoverable virtual shared disk (**rvsd**)

This command is typically run when a problem is encountered with any of these components in order to provide information to your software service organization.

The output of **ctsnap** is a compressed tar file (**ctsnap.***host_name***.***nnnnnnnn***.tar.Z**) and a log file (**ctsnap.***host_name***.***nnnnnnnn***.log**, where *nnnnnnnn* is the timestamp when the **ctsnap** command was run and *host_name* is the name of the host on which the command is running. Both files should be provided to the software service organization. By default, these files are placed in the **/tmp/ctsupt** directory.

Flags

–d *output_directory*
: Identifies the output directory. The default directory is **/tmp/ctsupt**.

–x runrpttr
Formats the trace file contents of all of the RSCT resource managers.

Using this flag will increase the size of the **ctsnap** output files, so you might need to increase the size of the file system that contains the output directory.

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

Files

/tmp/ctsupt Location of the default directory that contains the output files.

/tmp/ctsupt/ctsnap.*host_name***.***nnnnnnnn***.log**
Location of the log file of the command execution, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command is running.

tmp/ctsupt/ctsnap.*host_name***.***nnnnnnnn***.tar.Z**
Location of the compressed tar file that contains the collected data, where *nnnnnnnn* is a timestamp and *host_name* is the name of the host on which the command is running.

Exit status

0 Command has run successfully.

1 Command was not successful.

Security

Only **root** users can run this command.

Standard output

When the **-h** flag is specified, this command's usage statement is written to standard output.

Standard error

Error messages are written to standard error (and to the **ctsnap.***host_name***.***nnnnnnnn***.log** file).

Implementation specifics

This command is part of the Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/ctsnap Contains the **ctsnap** command

Examples

1. To gather RSCT support information, enter:

   ```
   ctsnap
   ```

2. To gather RSCT support information and place it in the **/tmp/mydir** directory, enter:

```
ctsnap -d /tmp/mydir
```

Related information

Commands: **lssrc**

Chapter 21. First failure data capture (FFDC) commands

© Copyright IBM Corp. 2002, 2007

fccheck

Purpose

Performs basic problem determination on the first failure data capture (FFDC) utilities.

Syntax

```
/usr/sbin/rsct/bin/fccheck [-q] | [-h]
```

Flags

-h

Displays help and usage information to standard output. No other processing is performed.

-q Specifies ″quiet″ mode. The command does not display the results of each test to standard output. The exit status of the command must be used to determine the results of the tests. If more than one condition was detected, the exit status will reflect the most severe condition detected by **fccheck**.

Description

The **fccheck** command performs basic problem determination for the first failure data capture utilities. The command checks for the following conditions and information on the local node:

- Checks if FFDC error stack usage has been disabled in the current process environment.
- Obtains the IP address that would be currently used by FFDC to identify the local node.
- Checks if **/var/adm/ffdc/stacks** is available, and if so, how much space is available in the file system where the directory resides. Checks to see if there is insufficient space to create FFDC error stacks.
- Checks if **/var/adm/ffdc/dumps** is available, and if so, how much space is available in the file system where the directory resides.

Results of these tests are displayed to standard output unless the ″quiet″ option has been specified. **fccheck** sets an exist status value to indicate the most severe condition it detected during the execution of its tests.

Exit status

0 All conditions tested by **fccheck** were found to be in normal operational parameters.

2 Help information successfully displayed. No further processing is performed.

12 No checking performed. Invalid option specified to this command.

19 The directory **/var/adm/ffdc/stacks** is not mounted or does not exist.

20 Cannot access or examine one or more directories in the path **/var/adm/ffdc/stacks**. Permissions may have been changed on one or more of the directories in this path to prevent access.

24 Cannot access or examine one or more directories in the path

/var/adm/ffdc/dumps. Permissions may have been changed on one or more of the directories in this path to prevent access.

32 The directory **/var/adm/ffdc/dumps** is not mounted or does not exist.

40 Insufficient space is available in the **/var/adm/ffdc/stacks** directory to create FFDC error stacks on the local node.

41 Unable to obtain file system information from the operating system. This indicates a potential problem with the operating system itself.

42 FFDC error stack creation and usage has been disabled in this process environment.

Examples

To check for possible problems with the FFDC utilities on the local node:

```
fccheck
fccheck Status: All tests completed
```

If the local node had disabled the creation of FFDC error stacks, **fccheck** would indicate this as a problem:

```
fccheck

fccheck Status: Creation and use of FFDC Error Stacks has been expressly
disabled in the current execution environment. Any processes created in
the current execution environment cannot create their own FFDC Error Stacks
or inherit use of existing FFDC Error Stacks.

fccheck Status: All checks completed. Examine the previous status output for
possible FFDC problem conditions and take the recommended actions listed in
these messages.
```

Implementation specifics

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

Related information

Commands: **fcclear**, **fcinit**

fcclear

Purpose

Removes first failure data capture (FFDC) error stacks and detail data files from the local node.

Syntax

```
/usr/sbin/rsct/bin/fcclear -h |
    {    [-d filename[,filename,...]]
         [-D filename[,filename,...]]
         [-f FFDC_failure_ID[,FFDC_failure_ID,...]
         [-F FFDC_failure_ID[,FFDC_failure_ID,...]
         [-s file_name[,filename,...]
         [-S file_name[,filename,...]
         [-t days]
    }
```

Flags

-d Removes detail data files by specifying a list of one or more detail data file names. These file names may be absolute path names, or relative to the **/var/adm/ffdc/dumps** directory. These files are removed if they exist on the local node. Files on remote nodes cannot be removed through this command. If more than one file name is provided, they must be separated by a comma (,) without any intervening white space.

-D

Preserves detail data files by specifying a list of one or more detail data file names. These file names may be absolute path names, or relative to the **/var/adm/ffdc/dumps** directory. These files are retained if they exist on the local node. Files on remote nodes cannot be retained through this command. If more than one file name is provided, they must be separated by a comma (,) without any intervening white space.

-f

Removes FFDC error stack files by specifying a list of one or more FFDC failure identifiers. The FFDC error stacks associated with these FFDC error identifiers are located and removed if they are present on the local node. FFDC error stacks on remote nodes will not be removed. If more than one FFDC failure identifier is supplied, they must be separated by a comma (,) with no intervening white space.

-F

Preserves FFDC error stack files by specifying a list of one or more FFDC failure identifiers. The FFDC error stacks associated with these FFDC error identifiers are located and retained if they are present on the local node. FFDC error stacks on remote nodes will not be retained. If more than one FFDC failure identifier is supplied, they must be separated by a comma (,) with no intervening white space.

-h

Displays help and usage information to the standard output device. No other processing is performed.

-s

Removes FFDC error stack files by specifying a list of one or more FFDC error stack file names. These file names can be absolute path names or file names relative to the **/var/adm/ffdc/stacks** directory. These files are removed if they exist on the local node. FFDC error stacks on remote nodes cannot be removed through this command. If more than one file name is provided, each must be separated by a comma (,) without any intervening white space.

-S

Removes FFDC error stack files by specifying a list of one or more FFDC error stack file names. These file names can be absolute path names or file names relative to the **/var/adm/ffdc/stacks** directory. These files are removed if they exist on the local node. FFDC error stacks on remote nodes cannot be removed through this command. If more than one file name is provided, each must be separated by a comma (,) without any intervening white space.

-t

Indicates that FFDC error stacks and detail data files that are older than a specific number of days should be removed from the local node. This selection criteria is independent of the other selection criteria.

Description

fcclear is used to remove FFDC error stack files that are no longer needed for problem determination efforts from the local node. Specific FFDC error stack files can be removed, as well as FFDC error stack files containing the records of specific FFDC failure identifiers. Individual entries within an FFDC error stack cannot be removed.

Using the **-t** option, **fcclear** can be used to remove FFDC error stack files older than a specific number of days. To use **fcclear** in an automatic fashion to clean out unneeded FFDC error stacks, see the **cron** command for automating the execution of commands.

To remove all FFDC error stacks from the local node, specify a value of zero (0) for the number of days option argument.

Exit status

fcclear generates the following exit status values upon completion:

0

Successful completion of the command. The command may complete successfully if no FFDC error stack files or detail data files match the selection criteria.

2

Help information successfully displayed. No further processing is performed.

10

No files are removed from the local system. A required option was not specified to this command.

11

No files are removed from the local system. The argument of the -t option is not numeric.

12

No files are removed from the local system. Unknown option specified by the caller.

19

The directory **/var/adm/ffdc/stacks** does not exist or is not mounted.

26

No files are removed from the local system. The same option was specified more than once.

28

No files were removed from the system. The caller provided options that instruct the command to both remove and retain the same file. This condition can occur when the command user specified an FFDC failure identifier that is recorded in an FFDC error stack file specified by name to this command.

Examples

To remove any FFDC error stack and detail data files older than seven days from the local node:

```
fcclear -t 7
```

To remove all FFDC error stack and detail data files older than seven days, but retain the FFDC error stack that contains information for the FFDC failure identifier **/3Iv04ZVVfvp.wtY0xRXQ7.....**, issue the command:

```
fcclear -t 7 -F /3Iv04ZVVfvp.wtY0xRXQ7...................
```

To remove the FFDC error stack file that contains the record for the FFDC failure identifier **/3Iv04ZVVfvp.wtY0xRXQ7.....**, issue the command:

```
fcclear -f /3Iv04ZVVfvp.wtY0xRXQ7...................
```

To remove the FFDC error stack files **myprog.14528.19990204134809** and **a.out.5134.19990130093256** from the system, plus the detail data file **myprog.14528.19990204135227**:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227
```

To extend the previous command to remove the named files plus any FFDC error stack and detail data files older that 14 days:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227 -t 14
```

Implementation specifics

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

Related information

Commands: **fccheck**, **fcreport**, **fcstkrpt**

fcdecode

Purpose

Translates a first failure data capture (FFDC) failure identifier from its standard form into its component parts and displays this information to standard output.

Syntax

```
/usr/sbin/rsct/bin/fcdecode FFDC_failure_ID[,FFDC_failure_ID,...] | -h
```

Flags

-h Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

Parameters

FFDC_failure_ID

An FFDC failure identifier, returned from previous calls to the **fcpushstk** and **fclogerr** commands, or returned from previous calls to the **fc_push_stack** or **fc_log_error** subroutines. This identifier indicates an entry made to report a failure or other noteworthy incident. More than one FFDC failure identifier can be provided as an argument to this command, however, each identifier must be separated by a comma (,) with no intervening white space between the identifiers.

Description

fcdecode decodes the 42-character FFDC failure identifier into its component parts, and displays these parts in human readable format. The output of this command displays the following information, extracted from the FFDC failure identifier:

- The network address (in ASCII format) of the node where this report resides
- The time when this recording was made, expressed using the currently active time zone settings
- One of the following, depending on where the information is recorded:
- The AIX error log template ID used to make this recording, if the record was filed in the AIX error log on that node, or
- The name of the FFDC error stack file containing this recording, if the record was file in the FFDC error stack and the FFDC error stack resides on this node
- A suggested command that can be used to obtain the specific report associated with this FFDC failure identifier.

Exit status

fcdecode returns one of the following integer status codes upon completion:

0

FFDC failure identifier successfully decoded.

2

Help information displayed and processing ended.

10

An FFDC failure identifier was not provided as an argument to this command.

12

Invalid or unsupported option provided to this command.

27

No information written to the standard output device. The FFDC failure identifier argument was not valid.

Examples

The FFDC failure identifier is represented by a base-64 value, read from right to left. Each dot represents a leading zero. To decode the FFDC failure identifier **.3Iv04ZVVfvp.wtY0xRXQ7.**.....into its component parts:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7...................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7...................
        Generated by the local system
        Generated Thu Sep 3 11:40:17 2003 EDT
        Recorded to the AIX Error Log using template 460bb505
        To obtain the AIX Error Log information for this entry, issue
        the following command on the local system:
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
        Search this output for an AIX Error Log entry that contains
        the following ERROR ID code:
         .3Iv04ZVVfvp.wtY0xRXQ7...................
```

The same command run on a different node has the following results:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7...................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7...................
        Generated on a remote system with the following Internet address:
          9.114.55.125
        Generated Thu Sep 3 11:40:17 2003 EDT
        Recorded to the AIX Error Log using template 460bb505
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
        Search this output for an AIX Error Log entry that contains
        the following ERROR ID code:
         .3Iv04ZVVfvp.wtY0xRXQ7...................
```

Implementation specifics

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

Related information

Commands: **fcdispfid**, **fcreport**, **fcstkrpt**

fcdispfid

Purpose

Displays the first failure data capture (FFDC) failure identifier to standard error.

Syntax

```
/usr/sbin/rsct/bin/fcdispfid [-q] FFDC_failure_ID | -h
```

Flags

-h Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-q

Suppresses warning messages from this command. If this option is not provided, this command will display messages when an invalid FFDC failure identifier is detected.

Parameters

FFDC_failure_ID

Specifies an FFDC failure identifier. This is an identifier returned from a previous call to **fcpushstk** or **fclogerr**, and indicates an entry made to report a failure encountered by the script. This identifier is written to the standard error device using FFDC message **2615-000**.

Description

This command is used by scripts to display an FFDC failure identifier value to the standard error device. This interface is provided because script programs do not have a mechanism for passing data back to its client except through exit status codes, signals, standard output, and standard error. To accomplish the task of ″passing back″ an FFDC failure identifier to a client in such an environment, **fcdispfid** uses XPG/4 cataloged message number **2615-000** to display this information to the standard error device. Clients of the script can capture the standard error information, search for the specific message number, and obtain the FFDC failure identifier from the script.

The script must indicate that any FFDC failure identifiers generated by the script will be directed to the standard error device in the script's user documentation. The client cannot be expected to know this behavior by default.

Exit status

0

FFDC failure identifier displayed to standard error.

2

Help information displayed and processing ended.

12

No information written to the standard error device. An invalid option was specified.

27

No information written to the standard error device. The *FFDC_failure_ID* argument does not appear to be in a valid format.

Examples

To display an FFDC failure identifier to the client through the standard output device:

```
FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
```

Implementation specifics

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

Related information

Commands: **fcdecode**, **fcfilter**, **fclogerr**, **fcpushstk**, **fcreport**, **fcstkrpt**

Subroutines: **fc_display_fid** in the *RSCT First Failure Data Capture Programming Guide and Reference*

fcfilter

Purpose

Locates and displays any first failure data capture (FFDC) failure identifiers in one or more files or in standard input.

Syntax

```
/usr/sbin/rsct/bin/fcfilter [ file_name ] [ . . . ]
```

Parameters

file_name

The name of the file to be searched for an FFDC failure identifier. More than one file may be provided. If a file name is not provided, **fcfilter** reads from standard input.

Description

This commands scans any files listed as arguments for FFDC failure identifiers. If a file name is not provided as an argument, this command examines standard input for FFDC failure identifiers. If an FFDC failure identifier is detected, **fcfilter** displays the identifier to standard output on its own line.

fcfilter can be used by scripts to extract FFDC failure identifiers returned by child processes via the standard error device.

If **fcfilter** detects more than one FFDC failure identifier in the input, the command will display all FFDC failure identifiers found, each one on a separate output line.

Exit status

fcfilter returns the following integer status codes upon completion:

0

fcfilter completed its execution. This exit status does not necessarily mean that any FFDC failure identifiers were detected.

> 0

fcfilter was interrupted or stopped by a signal. The exit status is the integer value of the signal that stopped the command.

Examples

The FFDC failure identifier is represented by a base-64 value, read from right to left. Each dot represents a leading zero. To obtain the list of all FFDC failure identifiers generated by a run of the command *mycmd*:

```
 mycmd 2> /tmp/errout
 fcfilter /tmp/errout
/.00...JMr4r.p9E.xRXQ7.....................
/.00...JMr4r.pMx.xRXQ7.....................
```

To obtain the FFDC failure identifier from a child process in a parent script, the script can use the **fcfilter** command as follows:

```
RESULTS=$(mychild 2> /tmp/errout)
if (($? != 0))              # mychild ended in failure, get FFDC ID
then
    cat /tmp/errout | fcfilter | read FIRST_FFDCID
else
    rm -f /tmp/errout
fi
```

Implementation specifics

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

Related information

Commands: **fcdispfid**, **fclogerr**, **fcpushstk**, **fcreport**, **fcstkrpt**

Subroutines: **fc_display_fid**, **fc_log_error**, **fc_push_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fcinit

Purpose

Establishes or inherits a first failure data capture execution environment.

Syntax

For Bourne and Korn shells:

```
. /usr/sbin/rsct/bin/fcinit.sh [ [-l] [-s{c|i}] ] | [-h]
```

For C shells:

```
source /usr/sbin/rsct/bin/fcinit.csh [ [-l] [-s{c|i}] ] | [-h]
```

Flags

-h Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-l Indicates that the process will use the AIX error log only. This option is not necessary when the **-s** option is specified, since use of the AIX error log is permitted within an FFDC error stack environment.

-s Indicates that an FFDC error stack environment is to be established. Applications wishing to use the **fcpushstk** interface must specify this flag. Upon successful completion of this command, an FFDC error stack file is reserved for the script in the **/var/adm/ffdc/stacks** directory. This flag must be specified with one of two possible options:

- **c** Requests that the FFDC error stack environment be *created*. If an FFDC error stack environment was not created by an ancestor process, it will be created. If such an environment was previously created by an ancestor process, this process will *inherit* the FFDC error stack environment as if the **i** option had been specified.
- **i** Specifies that an FFDC error stack environment is to be *inherited* if it was previously established by an ancestor process. If an FFDC error stack environment was not previously established by an ancestor process, an FFDC error stack environment is not established for this process, and this process cannot make use of an FFDC error stack (although it may make use of the AIX error log and the BSD system log).

Description

This interface must be used by a script program that will use the FFDC interfaces for recording information to the AIX error log, the BSD system log, or the FFDC error stack.

Applications may wish to establish an FFDC environment for one of the following reasons:

- The script may wish to record information to the AIX error log. Scripts can use **fcinit** to establish a basic FFDC environment
- The script wants to have itself and any descendant processes created by itself or its children to record failure information to the FFDC error stack. In this case, the script considers itself a ″top-level″ application that will cause multiple

″lower-level″ applications to be created, and the success of the ″top-level″ application depends upon the success of these ″lower-level″ applications. When using **fcinit** in this fashion, the process is said to *establish* or *create* the FFDC error stack environment.

- The script uses the FFDC error stack or the FFDC Trace only in those cases when the script is invoked by an ancestor process that wants failure information or trace information recorded to these devices. In all other cases, the script does not wish to use these devices. When using **fcinit** in this fashion, the process is said to *inherit* the FFDC error stack environment.

Any process wishing to record information to the AIX error log or the BSD system log through the FFDC interfaces must establish an FFDC environment. If the process does not wish to make use of an FFDC error stack, the process can establish a basic FFDC environment that does not make use of an FFDC error stack. An FFDC error stack environment, which contains an FFDC error stack, is established by a process when that process wants to have failure information from itself, any threads it may create, and any descendant processes it may create to be recorded in an FFDC error stack. An *FFDC error stack environment*, which contains an FFDC error stack, is inherited by a process when that process wants to record failure information to an FFDC error stack file only when one of its ancestors has requested for processes to do so; in all other cases, the process will not record failure information to the FFDC error stack.

The FFDC error stack environment, which contains an FFDC error stack, reserves an FFDC error stack file, so that failure information is recorded to a file in the **/var/adm/ffdc/stacks** directory. These files use the naming format ***script_name.PID.date_and_time***, where *script_name* is the name of the script itself, *PID* is the process identifier of the script, and *date_and_time* is the date and time when the script was executed. Whenever this script or children processes of this script record failure information to the FFDC error stack, it will be recorded in this file.

In order for information to be recorded in the FFDC error stack by a process, the process must use the **fcpushstk** FFDC interface, and the process has to be operating within an established FFDC error stack environment. If an FFDC error stack environment does not exist, or if the **fcpushstk** interface is not used when an FFDC error stack environment exists, no information is recorded by that process in the FFDC error stack. This function permits processes to run in a normal or ″silent″ mode when failure debugging information is not wanted or needed, but also permits this information to be available when the process is invoked within a special environment for debugging.

fcinit must be executed within the FFDC client's process environment (″sourced″) in order for the command to properly set the FFDC environment for the script. Script-based FFDC clients using this command must ″source″ the command in order for **fcinit** to execute within the client's process image. If this is not done. the FFDC interface is executed within its own process image; any settings of the FFDC environment are lost after the FFDC interface completes. To demonstrate how a script-based application would ″source″ the **fcinit** command, a Korn Shell program would issue the following instruction:

```
. fcinit.sh <options and arguments>
```

A C Shell script would do the following:

```
source fcinit.csh <options and arguments>
```

Processes that use the **fclogerr** FFDC interface must establish an *FFDC environment*. If the process only wishes to use the **fclogerr** interface, the *FFDC environment* can be established without an FFDC error stack.

If an FFDC environment already exists when a script attempts to create one, the script inherits the existing FFDC environment instead of creating its own.

Exit status

fcinit returns the following exit status codes upon completion:

0 FFDC environment successfully established.

1 FFDC environment successfully inherited.

2 Help information displayed and processing ended.

fcinit returns the following exit status codes upon detection of a failure:

12

FFDC environment not established or inherited - Unknown function parameter provided.

13

FFDC error stack environment not established or inherited - caller indicated that the FFDC environment should be both created and inherited.

14

FFDC environment not established in this call - the caller already has an FFDC environment established for itself - this routine may have been executed multiple times.

15

FFDC error stack environment not established or inherited - an FFDC error stack environment did not exist, and the FC_INHERIT option was specified.

16

FFDC environment not established or inherited - the client's process environment could not be modified by this routine.

17

FFDC environment not established or inherited - the FFDC environment appears to be corrupted and should be considered unusable.

18

FFDC environment not established or inherited - the routine could not allocate the memory required to modify the client's process environment.

19

FFDC error stack environment not established or inherited - Unable to reserve the FFDC error stack file for the calling process - the FFDC error stack directory does not exist or cannot be used.

21

FFDC error stack environment not established or inherited - Unable to reserve the FFDC error stack file for the calling process - the file already exists

42

FFDC error stack environment not established or inherited - creation and use of FFDC error stacks has been disabled by the system administrator. Scripts can establish only a basic FFDC environment that makes use of the AIX error log and the BSD system log.

99

FFDC environment not established or inherited - an unexpected internal failure occurred within **fcinit**. This condition may require the attention of customer and application-support services.

Examples

For a Korn Shell script to establish a basic FFDC environment for using the AIX error log and the BSD system log only (an FFDC error stack is not to be used or reserved):

```
# Set up an FFDC environment to use the AIX error log only. An FFDC error
# Stack is not needed for this script.
. fcinit.sh -1
rc=$?
if ((rc != 0))
    then
         print "fcinit failed with exit code of $rc"
         exit 1
fi
# Normal processing starts
```

For a Korn Shell script to establish an FFDC error stack environment that causes the script and any descendant process to record failure information to the FFDC error stack:

```
# Set up FFDC environment to record failure information to the FFDC error
# stack
. fcinit.sh -sc
rc=$?
if ((rc != 0))
then
    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

Note: The FFDC client may receive an indication that an FFDC error stack environment was inherited, instead of created by the **fcinit** call. This occurs when an FFDC error stack environment was already established by one of the process's ancestors.

To inherit an FFDC error stack environment from the process's parent process:

```
# Inherit an FFDC environment from parent process if it exists - otherwise,
# operate in a normal "silent" mode
. fcinit.sh -si
rc=$?
if ((rc != 0))
```

```
then
    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

Implementation specifics

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

Related information

Commands: **fccheck**, **fclogerr**, **fcpushstk**, **fcteststk**

Subroutines: **fc_init** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fclogerr

Purpose

Records information about failure or noteworthy conditions to the AIX error log and the BSD system log.

Syntax

```
/usr/sbin/rsct/bin/fclogerr {-e event -t error_template_label
        -i error_template_headerfile -r resource -s source_filename
        -p line_of_code_pos -v sidlevel -l lpp_name -a assoc_fid
        { [ -d detail_data_item[,detail_data_item,...]
        -x detail_data_type[,detail_data_type,...]
        -y detail_data_len[,detail_data_len,...] ] |
        [-f detail_data_file] } -b BSD_system_log_message_text } | -h
```

Flags

-a Contains the FFDC failure identifier for a failure condition reported by software used by this application which causes or influenced the condition being recorded at this time. This identifier should have been returned to this application as part of the software's result indication. The caller provides this identifier here so that the FFDC error stack can associate the failure report it is making at this time with the previously recorded failure report. This permits problem investigators to trace the cause of a failure from its various symptoms in this application and others to the root cause in the other software. If no other software failure is responsible for this condition, or if the other software did not return an FFDC failure identifier as part of its result information, this option should be omitted.

-b Specifies the text message to be written to the BSD system log.

-d Specifies one or more data items that provide detailed information about the condition, used to provide the detail data in the AIX error log entry. If details of the information are too lengthy, these details can be written to a file, and the name of that file provided as the *detail_data_file* parameter. If a detail data file name is provided, this option should be omitted. If neither the *detail_data* or the *detail_data_file* parameters are provided or appear valid, null information will be recorded for the detail data in the AIX error log.

More than one data item may be provided with this option. Each data item must be separated by commas (**,**) with no intervening white-space characters. If a data item has imbedded whitespace characters, the data item must be enclosed in double quotation marks (″ ″). The data items themselves must not contain commas (**,**), because the command interprets commands a field separators.

This option *must* be accompanied by the **-x** and **-y** options.

-e Specifies the FFDC log event type. Valid values are: **FFDC_EMERG**, **FFDC_ERROR**, **FFDC_STATE**, **FFDC_TRACE**, **FFDC_RECOV**, and **FFDC_DEBUG**. This code gives a general description of the type of event being logged (emergency condition, permanent condition, informational notification, debugging information, etc.) and the severity of the condition. If this option is not specified, the event type FFDC_DEBUG is assigned to this incident record.

-f Specifies the name of a file containing details about the condition being

reported. This option is used when the details are too lengthy to record within the remaining 100 bytes of detail data information left to the application by **fclogerr**, or when a utility exists that can analyze the detail information. The contents of this file is copied to the **/var/adm/ffdc/dumps** directory, and the file's new location is recorded as the detail data in the AIX error log entry.

-h Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-i Specifies the absolute path name of the header file (**.h**) that contains the AIX error log template identification number that corresponds to the *error_template_label* specified in the **-l** option. This template must also be found in the node's AIX error log template repository (**/var/adm/ras/errtmplt**). This header file was generated by the **errupdate** command as part of the source code's building procedures, and should have been included in the licensed program's packaging to be installed on the node with the software. If this option is not specified or the header file cannot be found when the script is executed, **fclogerr** will record the failure information using its own default error template (label FFDC_DEF_TPLT_TR, identifier code 2B4F5CAB).

-l Specifies an abbreviation of the name of the licensed program in which this software was shipped. This value should be recognizable to customer and application-support services as an acceptable name for the licensed program (**AIX** or **CSM**, for example). If this option is not provided or does not appear to be valid, the character string **PPS_PRODUCT** is used.

-p Specifies the line of code location within the source code module where the condition is being reported. The value provided must be a valid integer value. To allow for proper identification and location of the condition, this value should be as close to the line of code that detected the condition as possible. Korn Shell scripts can use the value of **$LINENO**. Script languages that do not provide a special line count variable can provide a symbolic value here that a developer can use to locate the spot in the source code where **fclogerr** is being used. If this option is not valid or not provided, the value of **0** is used.

-q Suppresses the generation of warning messages from the command. Warning are generated when the command must substitute default information for missing information, or when the command is unable to copy the *detail_data_file* to the **/var/adm/ffdc/dumps** directory.

-r Specifies the software component name. This is a symbolic name for the software making the report and should be a name recognizable to both customer and application-support services. The character string is limited to 16 characters.

-s Specifies the name of the source file containing the line of code that encountered the condition being reported. For Korn and Borne Shell scripts, the argument to this option should be set to **$0**; C Shell scripts would set this argument to **${0}**. If this option is not provided or not valid, the character string **unknown_file** is used.

-t Indicates the symbolic label given to the AIX error log template in the error log repository. The **errupdate** command that builds AIX error log templates creates a macro that maps this label to an integer code. This label begins with the characters **ERRID_** and is a maximum of 19 characters. If this

option is not specified or the header file cannot be found when the script is executed, **fclogerr** will invoke the **errlogger** to create a message in the AIX error log using the OPMSG template.

-v Indicates the SCCS version number of the source code module that detected the condition being recorded. For source code built under SCCS control, this should be set to ″**1.1**″ (the double-quotes are necessary). If this option is not provided or is not valid, the character string **unknown** is used.

-x Indicates how the data items specified by the **-d** option are to be interpreted when recording this information to the AIX error log. These types must agree with the corresponding fields of the AIX error log template specified in the **-t** option. Each type indicates how the corresponding data item in the **-d** list is interpreted. Acceptable values for this option are ALPHA, HEX, and DEC. There must be a matching type listed in the **-x** argument for each argument in the **-d** list.

This option *must* be supplied if the **-d** option is provided.

-y Indicates the length of the data items (in bytes) specified by the **-d** option. These lengths must agree with the corresponding fields of the AIX error log template specified in the **-t** option. There must be a matching type listed in the **-y** argument for each argument in the **-d** list.

This option *must* be supplied if the **-d** option is provided.

Description

This interface is used by any script program that wishes to record information to the AIX error log and the BSD system log. The information written to this device is intended for use by the system administrator or operator to determine what failure conditions or other noteworthy conditions have occurred on the system that require attention. The purpose of the AIX error log and the BSD system log is to record enough information about a condition so that the nature, impact, and response to the condition can be determined from the report, without requiring a recreation of the condition to detect what condition occurred and where. Any software that encounters permanent failure conditions that will persist until some type of direct intervention occurs, or encounters a condition that should be brought to the attention of the system administrator, should use **fclogerr** to record this information in the AIX error log and the BSD system log.

Scripts should establish a basic FFDC environment or an FFDC error stack environment before using **fclogerr**, either by creating or inheriting the environment. **fclogerr** records information to the AIX error log and the BSD system log even if these environments are not established, but the interface will not be capable of generating an FFDC failure identifier unless one of these environments exists.

Processes designed to use the FFDC error stack can also make use of the **fclogerr** interface, and should make use of it if they encounter conditions that require administrator attention or intervention to resolve.

To ensure proper identification of the condition and the location at which it was encountered, the FFDC policy recommends that **fclogerr** should be called in-line in the script's source code module and invoked as soon as the condition is detected. **fclogerr** will record source code file name and line of code information to assist in identifying and locating the source code that encountered the condition. **fclogerr** can be invoked by a subroutine or autoloaded routine to record this information if this is necessary, provided that all location information and necessary failure detail

information is made available to this external routine. The external recording routine must record the true location where incident was detected.

Although **fclogerr** reports information to both the AIX error log and the BSD system log, different options must be provided to this interface for each recording device. The detail data information recorded to the AIX error log is not recorded to the BSD system log. BSD system log information is provided through different command options. This may require the **fclogerr** user to duplicate some information in this call.

Exit status

fclogerr returns the following exit status codes upon successful completion:

0

Information successfully queued to be written to the AIX error log and the BSD system log. An FFDC failure identifier for the record is displayed to standard output. The caller should capture standard output to obtain this value.

2

Help information displayed and processing ended.

12

No information recorded to the AIX error log, and no FFDC failure identifier is provided by the command. The command user provided an non-valid option to this command.

On UNIX platforms other than AIX, **fclogerr** returns the following exit status codes when a failure occurs:

38 A record could not be made in the BSD system log for this incident. The system log is experiencing a failure condition. On AIX systems, a report was recorded to the AIX error log; on other systems, this should be considered a failure.

When **fclogerr** is provided with incomplete information, it substitutes default information for the missing information and attempts to make a record in the FFDC error stack. Warnings are generated in these cases, and warning messages are generated unless the **-q** option is specified. In cases where more than one warning condition was detected, the command returns an exit status code for the condition it considered the most severe. The following exit status codes are returned by **fclogerr** when warning conditions are detected:

10

The command user failed to provide the **-i** option to this command, or the header file named as the argument to the -i option could not be located. The command will record generic information to the AIX error log in this case, using the first failure data capture default template (label FFDC_DEF_TPLT_TR, identifier code 2B4F5CAB).

26

Both a detailed data string and a detail data file were provided to this routine. The routine chose the detail data string and ignored the detail data file.

28

The name of the resource detecting the incident was not provided. The default resource name **ffdc** was substituted for the missing resource name.

29

At least one component of the detecting application information—source code file name, source code file version, licensed program name, line of code position—was not provided. Default information was substituted for the missing information.

32

The file named in the *detail_data_file* parameter could not be copied to the **/var/adm/ffdc/dumps** directory. The FFDC error stack entry cites the original version of this file. Do not discard the original copy of this file.

33

The **-e** option was not specified, or did not specify a valid FFDC event type. The event type FFDC_DEBUG has been assigned to this incident record.

34

A message was not supplied in the *format* parameter. As a result, a generic message was recorded to the BSD system log for this incident.

35

No detailed information was provided for this incident. Later problem analysis may be difficult without these details to indicate specifics on the incident.

36

The length of the detail data string was greater than the capacity of the AIX error log entry limit. Detail data was truncated to fit in the available space. Some information on the incident may have been lost in this truncation.

37

An FFDC error identifier could not be constructed for the report created by this routine. An FFDC failure identifier is not written to standard output, but information on the incident was recorded to the AIX error log and the BSD system log.

38 A record could not be made in the BSD system log for this incident. The system log is experiencing a failure condition. On AIX systems, a report was recorded to the AIX error log; on other systems, this should be considered a failure.

Examples

For this example, a Korn Shell script attempts to access configuration information from a file. If this attempt fails, the code will record a failure to the AIX error log using the following template source code:

```
*! mymesgcat.cat
+ SP_FFDCEXMPL_ER:
    Comment        = "Configuration Failed - Exiting"
    Class          = S
    Log       = true
    Report         = true
    Alert          = false
    Err_Type  = PERM
    Err_Desc  = {3, 10, "CONFIGURATION FAILURE - EXITING"}
```

```
    Prob_Causes    = E89B
    User_Causes    = E811
    User_Actions   = 1056
    Fail_Causes    = E906, E915, F072, 108E
    Fail_Actions   = {5, 14, "VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE"},
               {5, 15, "VERIFY CONFIGURATION FILE"}
    Detail_Data    = 46, 00A2, ALPHA
    Detail_Data    = 42, EB2B, ALPHA
    Detail_Data    = 42, 0030, ALPHA
    Detail_Data    = 16, EB00, ALPHA
    Detail_Data    = 16, 0027, ALPHA
    Detail_Data    = 4, 8183, DEC
    Detail_Data    = 4, 8015, DEC
    Detail_Data    = 60, 8172, ALPHA
```

This definition yields the following AIX error log template:

```
LABEL:            ERRID_SP_FFDCEXMPL_ER
IDENTIFIER:        <calculated by errupdate during source code build>

Date/Time:         <filled in by AIX error log subsystem>
Sequence Number:   <filled in by AIX error log subsystem>
Machine Id:        <filled in by AIX error log subsystem>
Node Id:           <filled in by AIX error log subsystem>
Class:             S
Type:              PERM
Resource Name:     <filled in by -r option to fclogerr>

Description
CONFIGURATION FAILURE - EXITING

Probable Causes
COULD NOT ACCESS CONFIGURATION FILE

User Causes
USER CORRUPTED THE CONFIGRATION DATABASE OR METHOD

    Recommended Actions
    RE-CREATE FILE

Failure Causes
COULD NOT ACCESS CONFIGURATION FILE
PERMISSIONS ERROR ACCESSING CONFIGURATION DATABASE
FILE READ ERROR
FILE IS CORRUPT

    Recommended Actions
    VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE
    VERIFY CONFIGURATION FILE

Detail Data
DETECTING MODULE
<filled in by fclogerr options>
ERROR ID
<The FFDC Failure Identifier created by fclogerr>
REFERENCE CODE
<The -a option value to fclogerr>
FILE NAME
<Must be supplied as part of -d option list to fclogerr>
FUNCTION
<Must be supplied as part of -d option list to fclogerr>
RETURN CODE<Must be supplied as part of -d option list to fclogerr>
ERROR CODE AS DEFINED IN sys/errno.h
<Must be supplied as part of -d option list to fclogerr>
USER ID<Must be supplied as part of -d option list to fclogerr>
```

The first three **Detail_Data** fields are constructed by the **fclogerr** routine from information passed in the parameters. The remaining detail data must be supplied with the **-d** option, and the type of data supplied must be indicated by the **-x** option. The following sample source code segment demonstrates how this is done, and how **fclogerr** is invoked to record the information in the AIX error log and the BSD system log.

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="/configfile.bin"
exec 3< $CONFIG_FNAME
    :
read -u3 INBUF
RC=$?
if ((RC != 0))
then
    # Create detail data memory block for AIX error log template
    # Need to know the exact structure of the template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4-byte integer
    #    Field 7 - errno from failing function call (unused) - 4-byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the -d option, and
    # describes the data types for each in the -x option.
    MINUSDOPTS=$CONFIG_FNAME
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,read"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSDOPTS="$MINUSDOPTS,$RC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

Now consider a slight variation on the above example, using the same AIX error log template, but this time using an external command to obtain the configuration data from a file that this source code supplies. The command exits with a non-zero exit status and prints an FFDC failure identifier to standard output if it encounters any failure conditions. Also, to demonstrate the use of double quotation marks in the **-d** list, the configuration file will have an embedded space in the name:

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
typeset OUTPUT
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="This is a test"
OUTPUT=$(configdabeast $CONFIG_FNAME)
RC=$?
if ((RC != 0))
then
    # Create detail data memory block for AIX error log template
    # Need to know the exact structure of the template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4-byte integer
    #    Field 7 - errno from failing function call (unused) - 4-byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the -d option, and
    # describes the data types for each in the -x option.
    MINUSDOPTS="\""$CONFIG_FNAME"\""
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,configdabeast"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSDOPTS="$MINUSDOPTS,$RC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -a $OUTPUT -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

Implementation specifics

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

Related information

Commands: **errpt**, **fcdecode**, **fcdispfid**, **fcinit**, **fcpushstk**, **fcreport**

Files: **ct_ffdc.h**

Subroutines: **fc_log_error** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fcpushstk

Purpose

Records information about failure or noteworthy conditions to the first failure data capture (FFDC) error stack.

Syntax

```
/usr/sbin/rsct/bin/fcpushstk { [-aassoc_fid] -cmessage_catalog_name
    -mmessage_set -nmessage_number [-omessage_param[,message_param,...]]
    -llpp_name -pline_of_code_pos -rresource -ssource_filename -vsidlevel
    {[-ddetail_data] | [-fdetail_data_file]} } default_message | -h
```

Flags

-a Specifies an FFDC failure identifier for a failure condition reported by software used by this application which causes or influenced the condition being recorded at this time. This identifier should have been returned to this application as part of the software's result indication. The caller provides this identifier here so that the FFDC error stack can associate the failure report it is making at this time with the previously recorded failure report. This permits problem investigators to trace the cause of a failure from its various symptoms in this application and others to the root cause in the other software. If no other software failure is responsible for this condition, or if the other software did not return an FFDC failure identifier as part of its result information, the **-a** option should not be provided.

-c Indicates the name of the XPG/4-compliant message catalog that contains a description of the failure being recorded. This name is relative to the **/usr/lib/nls/msg/$LANG** directory. If the message catalog cannot be found, the *default_message* will be displayed to describe the failure. Note that the *default_message* will not be translated between locales.

-d A character string that provides detailed information on the condition, similar to the detail data concept used by the AIX error log. If details of the information are too lengthy, these details can be written to a file, and the name of that file provided as an argument to the **-f** option. The **-d** and **-f** options cannot be specified at the same time. If neither the **-d** or the **-f** options are provided or appear valid, the character string **no detail data** is recorded.

-f Specifies the name of a file containing details about the condition being reported, similar to the detail data concept used by the AIX error log. This option is used when the details are too lengthy to record within the FFDC error stack itself, or when a utility exists that can analyze the detail information. The contents of this file is copied to the **/var/adm/ffdc/dumps** directory, and the file's new location is recorded as the detail data in the FFDC error stack. If a file containing details of the condition does not exist, do not specify this option. The **-d** and **-f** options cannot be specified at the same time.

-h Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

-l Specifies an abbreviation of the name of the licensed program in which this software was shipped. This value should be recognizable to customer and application-support services as an acceptable name for the licensed

program (**AIX** or **CSM**, for example). If this option is not provided or does not appear to be valid, the character string **PPS_PRODUCT** is used.

-m Specifies the message set containing the message describing the failure in the message catalog file. If this message set cannot be located, the *default_message* will be displayed to describe the failure. Note that **default_message** will not be translated to the user's locale.

-n Specifies the message number that describes the failure being recorded. If this message cannot be located, the *default_message* will be displayed to describe the failure. Note that *default_message* will not be translated to the user's locale.

-o Specifies a list of substitution parameters within the message indicated by the **-n** option. **fcpushstk** only supports character strings as substitutional parameters (%s) due to the shell operating environment. If multiple substitutional parameters are provided, each one must be separated by a comma (,). If any of these substitution parameters contain imbedded white space, they must be enclosed in double quotes (″″).

-q Suppresses the generation of warning messages from the command. Warning are generated when the command must substitute default information for missing information, or when the command is unable to copy the *detail_data_file* to the **/var/adm/ffdc/dumps** directory.

-r Specifies the software component name. This is a symbolic name for the software making the report, and should be a name recognizable to both customer and application-support services.

-p Specifies the line of code location within the source code module where the condition is being reported. The value provided must be a valid integer value. To allow for proper identification and location of the condition, this value should be as close to the line of code that detected the condition as possible. Korn Shell scripts can use the value of **$LINENO**. Script languages that do not provide a special line count variable can provide a symbolic value here that a developer can use to locate the spot in the source code where **fcpushstk** is being used. If this option is not valid or not provided, the value of **0** is used.

-s Specifies the name of the source file containing the line of code that encountered the condition being reported. For Korn and Borne Shell scripts, the argument to this option should be set to **$0**; C Shell scripts would set this argument to ${0}. If this option is not provided or not valid, the character string **unknown_file** is used.

-v Indicates the SCCS version number of the source code module that detected the condition being recorded. For source code under SCCS control, this should be set to ″**1.1**″ (the double-quotes are necessary). If this option is not provided or is not valid, the character string **unknown** is used.

Parameters

default_message

Indicates a default message to be used as a description of the failure, when the information cannot be retrieved from the message catalog information supplied through the **-c**, **-m**, and **-n** options. If this string contains positional parameters, all positional parameters must be specified to be character strings (%s). The message should be enclosed in double quotes (″″) if it contains any embedded white space. **fcpushstk** limits the overall length of this string to 72 characters.

Description

fcpushstk is used by scripts to record failure information to the FFDC error stack. Scripts record descriptive information and debugging data to the FFDC error stack for use in later problem determination efforts.

The FFDC error stack is used to help understand failure conditions that occur when multiple related processes or threads are executing together on a node to perform a common task. This device is best applied to an application that creates one or more threads or subprocesses, which in turn, may also create threads or subprocesses themselves. To use the FFDC error stack, the script establishes an *FFDC error stack environment* using the **fcinit** interface. After this environment is established, the application and any of its descendants can make use of the FFDC error stack.

Not all software applications will establish an FFDC error stack environment. However, these applications may be invoked by other applications or scripts that establish FFDC error stack environments. In these cases, the scripts or applications invoking this software may wish to capture the failure information from this software, to analyze it along with other failure information from other software it invokes to discover any relationships or patterns in the failures. For this reason, software that ordinarily would not make use of the FFDC error stack under normal operational conditions should at least support the use of the FFDC error stack when it is used by any client invoking the software. This is accomplished by *inheriting* the FFDC error stack environment from the parent process through the **fcinit** interface.

fcpushstk records descriptions and details about noteworthy conditions to the FFDC error stack. If an *FFDC error stack environment* has not been established by the script, either by creation or inheritance, **fcpushstk** does not record any information and returns control back to the caller. This action permits the script to run in a normal ″silent″ mode when debugging information is not requested, but also permits the script to support the use of the FFDC error stack when debugging information is requested.

Scripts must make explicit calls to **fcpushstk** to record information to the FFDC error stack when an FFDC error stack environment is established. Merely establishing the environment is not enough to result in failure data being recorded. The **fclogerr** command will not make any records to the FFDC error stack.

To ensure proper identification of the condition and the location at which it was encountered, **fcpushstk** should be called in-line in the script's source code module, invoked as soon as the condition is detected. **fcpushstk** will record source code file name and line of code information to assist in identifying and locating the source code that encountered the condition. **fcpushstk** can be invoked by a subroutine or autoloaded routine to record this information if this is necessary, provided that all location information and necessary failure detail information is made available to this external routine. The external recording routine must record the true location where the incident was detected.

The maximum size of an FFDC error stack entry is given by the FC_STACK_MAX definition in the **<rsct/ct_ffdc.h>** header file. FC_STACK_MAX defines a length in bytes. This value should be used only as a rough guide, since this length includes data that will be used by **fcpushstk** to record the detecting file information, description information, and FFDC failure identifier information. Any records longer than FC_STACK_MAX bytes will be truncated to fit within the FC_STACK_MAX limit.

Exit status

fcpushstk returns the following exit status codes upon successful completion:

0

FFDC error stack environment exists, and failure information successfully recorded in the FFDC error stack. An FFDC failure identifier for the record is displayed to standard output. The caller should capture standard output to obtain this value.

2

Help information displayed and processing ended.

fcpushstk returns the following exit status codes when a failure occurs:

11

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. The client requested to use an option not supported in this release of the FFDC software

12

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. Unknown function parameter provided to the interface.

15

FFDC error stack environment does not exist. No information recorded to the FFDC error stack. No FFDC failure identifier is generated by this command. This is the normal return code to the FFDC client when an FFDC error stack environment did not exist to be inherited via **fcinit**.

17

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. The FFDC error stack environment appears to be corrupted and should be considered unusable.

19

No information recorded to the FFDC error stack - the FFDC error stack directory does not exist or cannot be used. No FFDC failure identifier is provided by this command.

20

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. Unable to access the FFDC error stack file. The file may have been removed, or permissions on the file or its directory have been changed to prohibit access to the FFDC error stack.

22

No information recorded to the FFDC error stack - the FFDC error stack file could not be locked for exclusive use by this interface. Repeated attempts had been made to lock this file, and all attempts failed. Another process may have locked the file and failed to release it, or the other process may be hung and is preventing other processes from using the FFDC error stack. No FFDC failure identifier is provided by this command.

24

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. The FFDC error stack file appears to be corrupted. The client should consider the FFDC error stack environment unusable.

25

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. The FFDC error stack file name is set to a directory name. The FFDC error stack environment should be considered corrupted and unusable.

32

A dump file could not be copied to the **/var/adm/ffdc/dumps** directory. There is insufficient space in the file system containing the **/var/adm/ffdc** directory. The **fcclear** command should be used to remove unneeded FFDC error stacks and dump files, or the system administrator needs to add more space to the file system. No FFDC failure identifier is provided by this command.

40

No information recorded to the FFDC error stack. Information could not be recorded in the FFDC error stack because there is not enough space in the file system that contains the **/var/adm/ffdc** directory. Use the **fcclear** command to remove unneeded FFDC error stacks and dump files. Otherwise, the system administrator needs to add more space to the file system. No FFDC failure identifier is provided by this command.

41

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. A failure occurred when reading control information from the FFDC error stack or writing incident information to the FFDC error stack. The client should conclude that the entry was not recorded for this incident.

99

No information recorded to the FFDC error stack, and no FFDC failure identifier is provided by this command. An unexpected internal failure occurred in the **fc_push_stack** routine.This problem may require the attention of application-support services.

When **fcpushstk** is provided with incomplete information, it substitutes default information for the missing information and attempts to make a record in the FFDC error stack. Warnings are generated in these cases, and warning messages are displayed to the standard error device unless the **-q** option has been specified. In cases where more than one warning condition was detected, the command generates an exit status code corresponding to the most severe warning condition it detected. The following exit status codes are returned by **fcpushstk** when warning conditions are detected:

26

Both a detailed data string and a detail data file were provided to this routine. The routine chose the detail data string and ignored the detail data file.

28

The name of the resource detecting the incident was not provided. The default resource name was substituted for the missing resource name.

29

At least one component of the detecting application information—source code file name, source code file version, licensed program name, line of code position—was not provided. Default information was substituted for the missing information.

30

No default message was provided to describe the nature of the incident. If the XPG/4 message catalog containing the description message cannot be found, no description for this condition will be displayed by the **fcstkrpt** command.

31

No message was provided to describe the nature of the incident, or a component of the XPG/4 information—catalog file name, message set number, message number—was not provided. No description for this condition can be displayed by the **fcstkrpt** command.

32

The file named in the *detail_data_file* parameter could not be copied to the **/var/adm/ffdc/dumps** directory. The FFDC error stack entry cites the original version of this file. Do not discard the original copy of this file.

35

No detailed information was provided for this incident. Later problem analysis may be difficult without these details to indicate specifics on the incident.

37

An FFDC failure identifier could not be constructed for the report created by this routine. No FFDC failure identifier is provided by this command, but information on the incident was recorded to the FFDC error stack.

44

The information provided to this command would have caused an FFDC error stack record to exceed the FC_STACK_MAX limit. The record was truncated to allow it to be recorded within the system limits. Important information about the failure may have been lost during the truncation process. Modify the script to provide less information, or to record the information to a detail data file and submit the detail data file name to this command instead.

Examples

To record information about a failure to the FFDC error stack when the FFDC environment is established or inherited by the process:

```
#!/bin/ksh
    :
    :
cp /tmp/workfile $FILENAME
RC=$?
if ((RC != 0))
then
```

```
    FFDCID=$(fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

To make the same recording from a script language that does not have a line of code variable available:

```
#!/bin/bsh
    :
    :
CODESCTN=14             # Used to identify where in the script code we are
cp /tmp/workfile $FILENAME
RC=$?
if test $RC -ne 0
then
    FFDCID=`fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$CODESCTN -v"1.1" -lPSSP "Cannot update configuration file %1$s"`
    if test $? -eq 0
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
CODESECTION=15               # New code section begins - a different task starts
    :
    :
```

To record information about a failure condition that is related to another failure condition previously recorded to the FFDC error stack by an application exploiting FFDC:

```
#!/bin/ksh
    :
    :
ASSOC_FID=$(/usr/lpp/ssp/bin/somecmd -a -b)
RC=$?if ((RC != 0))
then
    FFDCID=$(fcpushstk -a$ASSOC_FID -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then
         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

Implementation specifics

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

Related information

Commands: **fcdecode**, **fcdispfid**, **fcinit**, **fcreport**, **fcstkrpt**, **fcteststk**

Subroutines: **fc_push_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fcreport

Purpose

Locates and displays the report of a failure and any failures associated with the failure.

Syntax

```
/usr/sbin/rsct/bin/fcreport { [-a] FFDC_failure_ID } | -h
```

Flags

-a

Displays all information contained in a report for a failure. The default is to display the network address of the node where the failure report was generated, the time stamp on the failure report, and the description of the incident recorded in the failure report.

-h

Displays a help message to standard output and exits. No other processing is performed, regardless of the options specified.

Parameters

FFDC_failure_ID

Specifies the FFDC failure identifier of the failure to begin the report. **fcreport** will attempt to obtain the failure information for this failure, as well as any failures that this report lists as an associated failure. Only one FFDC failure identifier may be provided to this command.

Description

fcreport decodes an FFDC failure identifier, and obtains reports on the failure identified by it. The command also detects if any failure was associated with the FFDC failure identifier, and if so, obtains the report on that failure. The command continues to examine the report of each failure it locates for associated failures and to obtain reports on the associated failures until one of the following conditions is met:

- No further associated failures are detected.
- The report for an associated failure cannot be found. This may occur when the associated failure report resides on a remote node that cannot be reached at the moment, or the record of the failure has been removed from the node where it resided.

Using this command, the user can obtain a report for the entire list of failures that caused a specific failure. **fcreport** is not capable of locating reports for any failures that may have been caused by the initial failure provided to the command; it can only obtain reports of failures that caused this failure.

Exit status

fcreport generates one of the following exit status codes upon completion:

0

Failure report located and displayed for the FFDC failure identifier provided. Zero or more related failure reports may have been located and displayed as well.

2

Help information displayed and processing ended.

10

Required options or arguments are not provided.

11

The FFDC failure identifier provided to this command was generated by a later release of the FFDC software. The command is not capable of correctly interpreting this identifier.

12

Unknown option specified to this command.

20

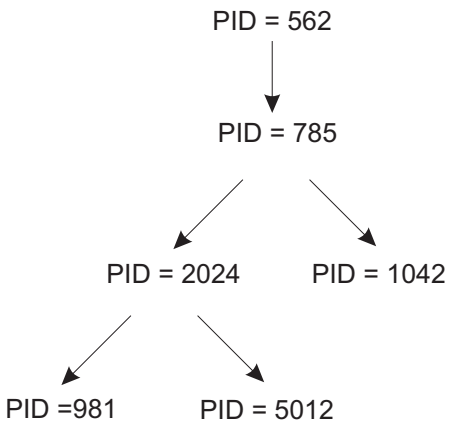
The FFDC failure identifier refers to an entry made in an FFDC error stack on this system, but the FFDC error stack file cannot be accessed. The file may have been removed, or permissions may have been altered on the file to prevent access to it.

27

The FFDC failure identifier provided to this command is not a valid identifier.

Examples

Consider the case where several processes were created in the following parent-child order:



In this example, process 785 generated the FFDC failure identifier **.3Iv04ZVVfvp.wtY0xRXQ7.....** and passed it back to Process 562. To obtain a detailed report for FFDC failure identifier **.3Iv04ZVVfvp.wtY0xRXQ7.....** and any previous failures that led to this specific failure:

```
$ fcreport -a .3Iv04ZVVfvp.wtY0xRXQ7...................
```

This report will contain the details of the specified FFDC failure identifier, as well as any failures in processes 2024, 1042, 981, and 5012 that may have caused it. The report will not contain any failures in process 562 that may have been caused as a result of process 785's failure.

Security

fcreport uses **rsh** to obtain failure reports that may reside on remote nodes. The user must have sufficient privilege to execute **rsh** commands to these remote nodes. If the user does not have this permission, **fcreport** can only trace the list of related failures so long as they exist on the local node.

Implementation specifics

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

Related information

Commands: **fcclear**, **fcdecode**, **fcdispfid**, **fcfilter**, **fclogerr**, **fcpushstk**, **fcstkrpt**

Subroutines: **fc_log_error**, **fc_push_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fcstkrpt

Purpose

Displays the contents of an FFDC error stack file.

Syntax

```
/usr/sbin/rsct/bin/fcstkrpt { [-a] [-p | -r]
                            {
                                 -fFFDC_failure_ID [-i] |
                                 -sFFDC_error_stack_file_name
                            }
                      } | [-h]
```

Flags

-a

Indicates that all information be displayed for entries in the FFDC error stack. The default action is to display the time stamp for the record and the description of the incident.

-f

Specifies the FFDC failure identifier to use in locating the FFDC error stack. **fcstkrpt** decodes the FFDC failure identifier, locates the FFDC error stack associated with that FFDC failure identifier, and processes the FFDC error stack. Only one FFDC failure identifier can be specified by this flag.

-h

Displays a help message to standard output and exits. No other processing is performed regardless of the options specified.

-i

Displays only the information associated with the specific failure report identified by the **-f** flag. By default, all records in the FFDC error stack are displayed.

-p

Displays information from the FFDC error stack by process orientation. The output is ordered so that it reflects the order in which the processes were created (parent-child process relationship). Child process information is shown first, followed by parent process information. This view is used to understand which incidents occurred first, and which incidents occurred later because of them.

-r

Displays information from the FFDC error stack by incident relationships. Incidents are presented along with those incidents that are related to them. This view is used to understand which incidents occurred because of the occurrence of other incidents. This is the default.

-s

Specifies the name of the FFDC error stack to be examined. This name may be either the absolute or relative path name of the FFDC error stack. Only one FFDC error stack file name can be specified by this flag. If a

relative file name is used, the file is assumed to be located in the **/var/adm/ffdc/stacks** directory of the node where the file resides.

Description

fcstkrpt reads an existing FFDC error stack file and displays its contents to the standard output device. The FFDC error stack file is indicated either through the name of the file itself, or by using an FFDC failure identifier that references a specific record within that file.

Information from the FFDC error stack can be displayed in one of two formats: by *related failure conditions* (the default) or by *software layer*.

Exit status

fcstkrpt issues the following integer exit status codes upon completion:

0

FFDC error stack file successfully located, and contents displayed to the standard output device.

2

Help information displayed and processing ended.

12

An invalid option was specified.

14

No information written to the standard output device. The **-f** option was used and the *FFDC error identifier* argument was not valid.

20

No information written to the standard output device. The **-s** option was used and the *FFDC error stack File* argument was not found.

27

No information written to the standard output device. The caller provided a valid *FFDC failure identifier*, but the file referenced by the FFDC failure identifier was not recorded on this node. Use the **fcdecode** command to locate the node where this FFDC error stack resides.

81

No information written to the standard output device. A failure occurred while writing information to standard output. The application should conclude that standard output cannot accept output.

85

No information written to the standard output device. The caller provided a valid FFDC failure identifier, but the file referenced by the FFDC failure identifier does not exist.

Examples

To obtain a brief report of the information stored in the FFDC error stack file **/var/adm/ffdc/stacks/myprog.562.20031001143052**:

```
$ fcstkrpt -r -s myprog.562.20031001143052
```

To obtain a detailed report of the information contained in the FFDC error stack where the FFDC failure identifier **.3Iv04ZVVfvp.wtY0xRXQ7.....** was recorded, and present this information in parent-child ordering:

```
$ fcstkrpt -p -f .3Iv04ZVVfvp.wtY0xRXQ7.....
```

Implementation specifics

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

Related information

Commands: **fcclear**, **fcdecode**, **fcdispfid**, **fcfilter**, **fcpushstk**, **fcreport**

Subroutines: **fc_push_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

fcteststk

Purpose

Tests for the presence of a first failure data capture error stack environment.

Syntax

```
/usr/sbin/rsct/bin/fcteststk [-q] | [-h]
```

Flags

-q

Suppresses output from this command that explains whether or not an FFDC environment was established. The command user will be required to test the exit status from the command to determine whether an FFDC environment is established for this process.

-h Writes this command's usage statement to standard output.

Description

fcteststk can be called by any application program that wishes to use the FFDC error stack to test if these facilities have been activated . By performing this test, applications can avoid the performance burden of collecting failure information in cases where an *FFDC environment* has not been established. This interface is provided primarily for use by library routines, which would not have any knowledge of whether their client application established or inherited an *FFDC environment*.

An *FFDC error stack environment* is established by a process when that process wants to have failure information from itself, any threads it may create, and any descendant processes it may create to be recorded in an FFDC error stack. An *FFDC error stack environment* is inherited by a process when that process wants to record failure information to an FFDC error stack file only when one of its ancestors has requested for processes to do so; in all other cases, the process will not record failure information to the FFDC error stack. Processes use **fcinit** to either establish or inherit the FFDC error stack environment.

The FFDC error stack environment reserves an FFDC error stack file, so that failure information is recorded to a file in the **/var/adm/ffdc/stacks** directory. These files use the naming format ***script_name.PID.date_and_time***, where *script_name* is the name of the script itself, *PID* is the process identifier of the script, and *date_and_time* is the date and time when the script was executed. Whenever this script or children processes of this script record failure information to the FFDC error stack, it will be recorded in this file.

Applications use the **fcpushstk** interface to record failure information to the FFDC error stack. However, the application may need to collect this information from various locations before recording the information, and obtaining this information can impact the application's overall performance. The application should not need to collect this information if the *FFDC error stack environment* was not established or inherited. To avoid this performance impact, the application can issue **fcteststk** to determine if an *FFDC error stack environment* is available, and if so, begin collecting the failure information. If the *FFDC error stack environment* does not exist, the application can avoid collecting this information.

Processes that use the **fclogerr** FFDC interface can use **fclogerr** when an *FFDC environment* exists, whether or not an FFDC error stack is in use by the *FFDC environment*. Whenever **fclogerr** is used, failure information is recorded to the AIX error log and the BSD system log, regardless of whether an FFDC error stack was reserved. Any application that records information using the **fclogerr** interface must *always* collect the failure information and record it, regardless of whether an FFDC error stack is in use.

Exit status

0 An FFDC error stack environment exists.

2 Help information displayed and processing ended.

12 No processing performed. An invalid option was specified.

15 FFDC error stack environment has not been established or inherited by the client at this point in time.

17 FFDC error stack environment appears to be corrupted and should be considered unusable.

Examples

To test whether an FFDC error stack environment exists for an application:

```
fcteststk -q
if (($? == 0))
then
    # Collect failure information
         :
         :
    # Use fcpushstk to record failure info
         :
         :
fi
```

Implementation specifics

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

Related information

Commands: **fcinit**, **fcpushstk**

Subroutines: **fc_test_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

Chapter 22. FFDC files

© Copyright IBM Corp. 2002, 2007

ct_ffdc.h

Purpose

Provides data types, definitions, and interface prototypes for the first failure data capture (FFDC) C language library interfaces.

Description

Any C and C++ language source code files that use the FFDC C language interfaces must include this header file. This file contains the C language prototypes for the FFDC interfaces, the symbolic constants used as return codes from these interfaces, and data type definitions needed by FFDC C and C++ language clients.

C language interface selection control

This file provides the compiler definition **FC_VERSION**. This definition controls which version of the FFDC interfaces should be used during compilation of the source code. One version of the FFDC interfaces is available; the value of **FC_VERSION** is set to a default value of **1**. If this variable is not set during compilation, the value for **FC_VERSION** reverts to the default value of **1** and the initial version of the FFDC interfaces is used.

Data types

The **fc_eid_t** data type defined by this module is used to store an FFDC failure identifier. This identifier is created by the **fc_push_stack** and **fc_log_error** interfaces whenever these interfaces are successful in recording failure information. This identifier contains information in an encoded form to indicate the system on which the record was made, the time when the record was made, and the location of the record. FFDC commands such as **fcreport** and **fcstkrpt** can be used at a later time to obtain the exact failure report for problem determination efforts.

FFDC environment establishment codes

An FFDC client application uses the **fc_init** interface to specify how the FFDC environment should be established. The following selections are supported:

FC_LOG
: Establishes a basic FFDC environment, which permits the application to record failure information to the AIX error log and the BSD system log. An FFDC error stack is not established for use by the application in this case, unless this value is combined with the FC_STACK_CREAT option or the FC_STACK_INHERIT option. This selection would be used by applications making use of the **fc_log_error** interface only.

FC_STACK_CREAT
: Creates an FFDC error stack environment if one was not previously created by an ancestor of this process, or inherits the FFDC error stack environment if an ancestor previously established one. The FFDC error stack environment permits the application to record information to the AIX error log, the BSD system log, and the FFDC error stack. This selection is used by applications that would use the **fc_push_stack** interface as well as the **fc_log_error** interface. Do not use FC_STACK_CREAT in conjunction with FC_STACK_INHERIT.

FC_STACK_INHERIT
: Inherits an FFDC error stack environment only if an ancestor process previously established an FFDC error stack environment. If an ancestor did not establish such an environment, the application does not make use of an

FFDC error stack, but may still make use of the AIX error log and the BSD system log. Do not use FC_STACK_INHERIT in conjunction with FC_STACK_CREAT.

Record type definitions

Seven FFDC event types are specified in this file. These event types are used to instruct the **fc_log_error** interface as to the severity of the condition being logged:

FFDC_EMERG
A severe failure has occurred, and the system is in danger of coming offline. This information is required by the system administrator to bring the system back online. The AIX error log type equivalent is PEND. The BSD system log priority equivalent is LOG_EMERG.

FFDC_ERROR
A permanent failure has occurred, and the condition will persist until it is repaired. The system is not in danger of coming offline. The AIX error log type equivalent is PERM. The BSD system log priority equivalent is LOG_ERR.

FFDC_STATE
An event of some significance has occurred, but the event does not constitute a failure condition. The AIX error log type equivalent is INFO. The BSD system log priority equivalent is LOG_NOTICE.

FFDC_PERF
A condition has been noticed which can or will degrade the system's performance below acceptable levels. The system is not in danger of coming offline, but performance may be unacceptably slow, which can result in random failures in system applications, such as timeout conditions. The AIX error log type equivalent is PERF. The BSD system log priority equivalent is LOG_WARNING.

FFDC_TRACE
This entry identifies the name and location of a trace file generated by an application or system component. Such an entry would be made when a trace has been enabled in an application or a component, to indicate where the trace file resides. The AIX error log type equivalent is UNKN. The BSD system log priority equivalent is LOG_INFO.

FFDC_RECOV
A recovery action has been successfully completed by the system in response to an FFDC_EMERG, FFDC_ERROR, or FFDC_PERF condition. Such an entry would be created only after an FFDC_EMERG, FFDC_ERROR, or FFDC_PERF condition was detected, and a recovery action started in response to that condition completed successfully. The AIX error log type equivalent is TEMP. The BSD system log priority equivalent is LOG_DEBUG.

FFDC_DEBUG
A failure condition was detected. Unlike the FFDC_ERROR case, the failure is not a permanent condition, or the system can continue successfully with the condition present. The AIX error log type equivalent is UNKN. The BSD system log priority equivalent is LOG_DEBUG.

Location

/usr/sbin/rsct/include/ct_ffdc.h

/usr/include/rsct/ct_ffdc.h

Related information

Commands: **fcdecode**, **fcdispfid**, **fcreport**, **fcstkrpt**

Subroutines: **fc_display_fid**, **fc_eid_init**, **fc_eid_is_set**, **fc_init**, **fc_log_error**, **fc_push_stack** (see the *RSCT First Failure Data Capture Programming Guide and Reference*)

Examples

To use this file in a C or C++ language program, add the following line near the beginning of the source code module:

```
#include <rsct/ct_ffcd.h>
```

Part 11. The event management subsystem

Chapter 23. Event management commands 597
emsvcsctrl 598
haemd 602
haemd_HACMP 603
haemqvar 604
haemtrcoff 608
haemtrcon 610
haemunlkrm 612

© Copyright IBM Corp. 2002, 2007

Chapter 23. Event management commands

© Copyright IBM Corp. 2002, 2007

emsvcsctrl

Purpose

Starts the event management subsystem.

Syntax

emsvcsctrl [**–a** | **–s** | **–k** | **–d** | **–c** | **–t** | **–o** | **–h**]

Flags

-a Adds the subsystem.

-s Starts the subsystem.

-k Stops the subsystem.

-d Deletes the subsystem.

-c Cleans the subsystem.

-t Turns tracing on for the subsystem.

-o Turns tracing off for the subsystem.

-h Displays usage information.

Description

emsvcsctrl is a control script that starts the event management subsystem. Event management is a distributed subsystem of RSCT that provides a set of high-availability services for the IBM RS/6000® server. By matching information about the state of system resources with information about resource conditions that are of interest to client programs, it creates events. Client programs can use events to detect and recover from system failures, thus enhancing the availability of the system. The emsvcsctrl control script controls the operation of the Event Management subsystem. The subsystem is under the control of the System Resource Controller (SRC) and belongs to a subsystem group called emsvcs. A daemon is associated with each subsystem. The emsvcsctrl script also controls the operation of the AIX Resource Monitor subsystem. The subsystem is under SRC control and also belongs to the emsvcs subsystem group. A daemon is associated with each subsystem.

Instances of the Event Management and AIX Resource Monitor subsystems execute on each node in the HACMP/ES cluster. From an operational point of view, the Event Management subsystem group is organized as follows:

Subsystem
Event Management

Subsystem Group
emsvcs

SRC Subsystem
The emsvcs subsystem is associated with the haemd daemon.

emaixos
The emaixos is associated with the harmad daemon.

Daemons
The haemd daemon provides the Event Management services. The harmad daemon is the resource monitor for AIX operating system resources.

The emsvcsctrl script is not normally executed from the command line. It is normally called by the HACMP/ES startup script command during installation of the system.

The emsvcsctrl script provides a variety of controls for operating the Event Management subsystem:

- Adding, starting, stopping, and deleting the subsystem
- Cleaning up the subsystems
- Turning tracing on and off

Adding the Subsystem: When the -a flag is specified, the control script uses the mkssys command to add the Event Management and AIX Resource Monitor subsystems to the SRC. The control script operates as follows:

1. It makes sure that the emsvcs and emaixos subsystems are stopped.
2. It removes the emsvcs and emaixos subsystems from the SRC (just in case they are still there).
3. It adds the emsvcs subsystem to the SRC.
4. It adds the emaixos subsystem to the SRC.
5. It adds haemrm group using the mkgroup command, if it does not already exist. Any errors that occur are written to a log file named /var/ha/log/em.mkgroup.
6. It creates the /var/ha/lck/haem and /var/ha/soc/haem directories, if they don't already exist. Any errors that occur are written to a log file named /var/ha/log/em.mkdir.
7. It copies the Event Management Configuration Database, (EMCDB) from its install location, /usr/sbin/rsct/install/config/em.HACMP.cdb to its run-time location, /etc/ha/cfg/em.HACMP.cdb. Any errors resulting from the copy are written to a log file named /var/ha/log/em.cp.

Starting the Subsystem: When the -s flag is specified, the control script uses the startsrc command to start the Event Management subsystem, emsvcs, and the AIX Resource Monitor subsystem, emaixos.

Stopping the Subsystem: When the -k flag is specified, the control script uses the stopsrc command to stop the Event Management subsystem, emsvcs, and the AIX Resource Monitor subsystem, emaixos.

Deleting the Subsystem: When the -d flag is specified, the control script uses the rmssys command to remove the Event Management and AIX Resource Monitor subsystems from the SRC. The control script operates as follows:

1. It makes sure that the emsvcs and emaixos subsystems are stopped.
2. It removes the emsvcs and emaixos subsystems from the SRC using the rmssys command.

Cleaning Up the Subsystems: When the -c flag is specified, the control script stops and removes the Event Management subsystems for all system partitions from the SRC. The control script operates as follows:

1. It stops all instances of subsystems in the subsystem group by using the stopsrc -g emsvcs command.
2. It removes all instances of subsystems in the subsystem group from the SRC using the rmssys command.
3. It removes the Event Management Configuration Database (EMCDB) from its run-time location, /etc/ha/cfg/em.HACMP.cdb.

Turning Tracing On: When the -t flag is specified, the control script turns tracing on for the haemd daemon, using the haemtrcon command. Tracing for the harmad daemon is also enabled, using the traceson command.

Turning Tracing Off: When the -o flag is specified, the control script turns tracing off for the haemd daemon, using the haemtrcoff command. Tracing for the harmad daemon is also disabled, using the tracesoff command.

Logging: While it is running, the Event Management daemon normally provides information about its operation and errors by writing entries to the AIX error log. If it cannot, errors are written to a log file called /var/ha/log/em.default.cluster_name.

Files

/var/ha/log/em.default.cluster_name
Contains the default log of the haemd daemon on the cluster named cluster_name.

/var/ha/log/em.cp Contains a log of any errors that occurred while copying the Event Management Configuration Database.

/var/ha/log/em.trace.cluster_name
Contains the trace log of the haemd daemon on the cluster named cluster_name.

/var/ha/log/em.mkgroup Contains a log of any errors that occurred while creating the haemrm group.

/var/ha/log/em.mkdir Contains a log of any errors that occurred while creating the /var/ha/lck/haem and /var/ha/soc/haem directories.

Standard error

This command writes error messages (as necessary) to standard error.

Exit status

0 Indicates the successful completion of the command.

1 Indicates that an error occurred.

Security

You must be running with an effective user ID of **root**.

Restrictions

This command is valid in an HACMP environment only.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/emsvcsctrl Contains the **emsvcsctrl** script

Examples

1. To add the Event Management subsystem to the SRC, enter:
   ```
   emsvcsctrl -a
   ```
2. To start the Event Management subsystem, enter:
   ```
   emsvcsctrl -s
   ```
3. To stop the Event Management subsystem, enter:
   ```
   emsvcsctrl -k
   ```
4. To delete the Event Management subsystem from the SRC, enter:
   ```
   emsvcsctrl -d
   ```
5. To clean up the Event Management subsystem, enter:
   ```
   emsvcsctrl -c
   ```
6. To turn tracing on for the Event Management daemon, enter:
   ```
   emsvcsctrl -t
   ```
7. To turn tracing off for the Event Management daemon, enter:
   ```
   emsvcsctrl -o
   ```

Related information

Commands: **haemtrcoff**, **haemtrcon**, **lssrc**, **startsrc**, **stopsrc**

Daemons: **haemd**

haemd

Purpose

Observes resource variable instances that are updated by Resource Monitors and generates and reports events to client programs.

Syntax

haemd

Description

The **haemd** (Event Manager) daemon observes resource variable instances that are updated by Resource Monitors and generates and reports events to client programs.

One instance of the haemd daemon executes on every node of a cluster. The haemd daemon is under System Resource Controller (SRC) control.

Because the daemon is under SRC control, it cannot be started directly from the command line. It is normally started by the emsvcsctrl command. If you must start or stop the daemon directly, use the emsvcsctrl command.

When SRC creates the haemd daemon, the actual program started is haemd_HACMP. The haemd_HACMP program, after collecting information needed by the daemon, then executes the haemd program. In other words, the **haemd_HACMP** program is replaced by the **haemd** program in the process created by SRC.

For more information about the Event Manager daemon, see the **emsvcsctrl** command.

Implementation specifics

This daemon is part of Reliable Scalable Cluster Technology (RSCT) fileset for AIX.

Location

/usr/sbin/rsct/bin/haemd Location of the **haemd** daemon

Related information

emsvcsctrl, **haemd_HACMP**

haemd_HACMP

Purpose

Start-up program for the Event Manager daemon.

Syntax

haemd_HACMP [**-d** *trace_arg*]

Flags

-d *trace_arg*
: Should only be used under the direction of the IBM Support Center. The possible trace arguments are the same as for the **haemtrcon** command, except for **reg** and **dinsts**. To use this flag, the **emsvcs** subsystem definition in the system resource controller (SRC) must be changed using the **chssys** command with the **-a** flag. The daemon must then be stopped and restarted.

Description

The **haemd_HACMP** program is the startup program for the **haemd** daemon. When the event management subsystem is configured in the SRC by the **emsvcsctrl** command, **haemd_HACMP** is specified as the program to be started.

This program can only be invoked by the SRC. To start the event management subsystem, use the **emsvcsctrl** command.

Restrictions

This command is valid in an HACMP environment only.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/haemd_HACMP
: Location of the **haemd_HACMP** program

Related information

emsvcsctrl, haemd, haemtrcon

haemqvar

Purpose

Queries resource variables.

Syntax

haemqvar [**-H** *domain* | **-S** *domain*] [**-c** | **-d** | **-i**] [**-f** *file*] [**-h**] [*class var rsrcID* [*″*]]

Flags

-H *domain*
: Queries resource variables in the HACMP domain specified by *domain*.

-S *domain*
: Queries resource variables in the RS/6000 SP domain specified by *domain*.

-c Queries current resource variable values.

-d Queries resource variable definitions but produces short form output.

-i Queries instances of resource variables.

-f *file* Queries resource variables specified in *file*.

-h Displays a usage statement.

Parameters

class Specifies the name of the resource variable class or a null string.

var Specifies the name of the resource variable or a null string.

rsrcID Specifies a resource ID or an asterisk.

Description

The haemqvar command queries the Event Management subsystem for information about resource variables. By default, the command writes to standard output the definitions for all resource variables in the current SP domain, that is, the current SP system partition as defined by the SP_NAME environment variable. If SP_NAME is not set the default system partition is used. The -S flag can be used to specify another SP domain (system partition). To query variables in an HACMP domain, use the -H flag. For an SP domain, the domain flag argument is a system partition name. For an HACMP domain, the domain flag argument is an HACMP cluster name. When the -H flag is specified, the command must be executed on one of the nodes in the HACMP/ES cluster.

The following information is reported for each resource variable definition:

- Variable Name
- Value Type
- Data Type
- SBS Format (if data type is Structured Byte String)
- Initial Value
- Class
- Locator
- Variable Description

- Resource ID and its description
- Default Expression (if defined) and its description

Because the default behavior of this command can produce a large amount of output, standard output should be redirected to a file.

If the -d flag is specified only the resource variable name and a short description are written to standard output, one name and description per line.

If the -c flag is specified the current values of all resource variables instances are written to standard output, one per line. The line of output contains the location of the resource variable instance (node number), the resource variable name, the resource ID of the instance and the resource variable instance value. If the resource variable is a Structured Byte String (SBS) data type, then the value of each SBS field is reported.

The -i flag reports the same information as the -c flag except that the value of the variable instance is the last known value rather than the current value. The -i flag is useful for determining what resource variable instances exist.

For both the -c and the -i flags, if an error is encountered in obtaining information about a resource variable instance, the output line contains an error message, symbolic error codes, the location of where the error originated (if it can be determined), the resource variable name and the resource ID.

To return information about specific resource variables, specify the class, var and rsrcID operands. These operands can be repeated to specify additional resource variables. In addition, the var and rsrcID operands can be wildcarded to match a number of resource variables. Note that null string operands or an asterisk must be quoted in the shells.

If class is not a null string, then all variables in the specified class, as further limited by the var and rsrcID arguments, are targets of the query. If class is a null string, then variables of all classes, as further limited by the var and rsrcID arguments, are targets of the query. The var argument can be wildcarded in one of two ways:

1. Specify the variable name as a null string
2. Truncate the name after any component

When the resource variable name is wildcarded in the first manner, then all resource variables, as further limited by the class and rsrcID arguments, are targets of the query. When the resource variable name is wildcarded in the second manner, all resource variables whose high-order (leftmost) components match the var argument, as further limited by the class and rsrcID arguments, are targets of the query.

All resource variable instances, or definitions if neither the -c nor the -i flags are specified, of the variables specified by the class and var arguments that match the rsrcID argument are the targets of the query.

If neither the -c nor the -i flags are specified, the rsrcID argument is a semicolon-separated list of resource ID element names. If either the -c or the -i flags is specified, the rsrcID argument is a semicolon-separated list of name/value pairs. A name/value pair consists of a resource ID element name followed by an equal sign followed by a value of the resource ID element. An element value may consist of a single value, a range of values, a comma-separated list of single values

or a comma-separated list of ranges. A range takes the form a-b and is valid only for resource ID elements of type integer (the type information can be obtained from the variable definition). There can be no blanks in the resource ID.

A resource ID element is wildcarded by specifying its value as the asterisk character. Only variables that are defined to contain the elements, and only the elements, specified in the rsrcID argument are targets of the query. If any element of the resource ID consists of the asterisk character, rather than a name/value pair (or just a name if querying for definitions), all variables that are defined to contain at least the remaining specified elements are targets of the query. The entire resource ID is wildcarded if it consists of only the asterisk character; all instances of all resource variables, as further limited by the class and var arguments, are targets of the query.

Note that the rsrcID argument must be quoted in the shells if it contains semicolons or asterisks.

The class, var and rsrcID operands can be placed in a file, one set of operands per line, instead of being specified as command arguments. Use the -f flag to specify the name of the file to the command. If the -f flag is used, any operands to the command are ignored. Within the file, null strings are specified as two adjacent double quotation marks. A completely wildcarded resource ID can either be a single asterisk (*) or an asterisk in double quotation marks (″*″). The arguments must be separated by blank spaces or tabs on each line.

Some examples of using wildcards in the rsrcID argument follow. For these examples, assume the class and var arguments are null strings. If either the class or var arguments or both are not null strings, targets for the query are restricted accordingly. In the first three examples, all variables whose resource IDs are defined to contain the elements NodeNum, VG and LV, and only those elements, are matched.

1. In this example, only one instance is matched:

   ```
   NodeNum=5;VG=rootvg;LV=hd4
   ```

2. In this example, one instance from each node is matched:

   ```
   NodeNum=*;VG=rootvg;LV=hd4
   ```

3. In this example, all instances of the matching resource variables are matched:

   ```
   NodeNum=*;VG=*;LV=*
   ```

4. In this example, all variables whose resource IDs are defined to contain only the element NodeNum are matched. The instances matched are associated with node 9:

   ```
   NodeNum=9
   ```

5. In this example, the same set of variables are matched, but all instances of each variable are matched:

   ```
   NodeNum=*
   ```

6. In this example, all variables whose resource IDs are defined to contain elements NodeNum and VG, as well as zero or more additional elements, are matched. The instances matched are associated with node 9:

   ```
   NodeNum=9;VG=*;*
   ```

7. In this example, all variables whose resource IDs are defined to contain the element NodeNum, as well as zero or more additional elements, are matched. All instances of the variables are matched:

   ```
   NodeNum=*;*
   ```

Given the flexibility in specifying resource variables for query, it is possible that no resource variable instance or resource variable definition will match. If there is no match appropriate error information is reported, either in the form described above or as follows.

If the specification of the class, var or rsrcID arguments are in error, the output line contains an error message, symbolic error codes and the specified class name, resource variable name, and resource ID.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/haemqvar Location of the **haemqvar** command

Examples

1. To obtain the definitions of all resource variables in the current cluster and place the output in a file, enter:

   ```
   haemqvar -H HAcluster > vardefs.out
   ```

2. To obtain a short form list of all resource variables whose resource IDs contain the element VG, in the HACMP cluster named HAcluster, enter:

   ```
   haemqvar -H HAcluster -d "" "" "VG;*"
   ```

3. To obtain resource variables whose resource IDs contain only the elements VG and NodeNum, enter:

   ```
   haemqvar -H HAcluster -d "" "" "VG;NodeNum"
   ```

haemtrcoff

Purpose

Turns tracing off for the Event Manager daemon.

Syntax

haemtrcoff -s *subsys_name* **-a** *trace_list*

Flags

-s *subsys_name*
: Specifies the name of the Event Management subsystem. On a node this is emsvcs. This argument must be specified.

-a *trace_list*
: Specifies a list of trace arguments. Each argument specifies the type of activity for which tracing is to be turned off. At least one argument must be specified. If more than one argument is specified, the arguments must be separated by commas. The list may not include blanks.

Parameters

The following trace arguments can be specified:

init Stops tracing the initialization of the Event Manager daemon.

config Stops dumping information from the configuration file.

insts Stops tracing resource variable instances that are handled by the daemon.

rmctrl Stops tracing Resource Monitor control.

cci Stops tracing the client communication (internal) interface.

emp Stops tracing the event manager protocol.

obsv Stops tracing resource variable observations.

evgn Stops tracing event generation and notification.

reg Stops tracing event registration and unregistration.

pci Stops tracing the peer communication (internal) interface.

msgs Stops tracing all messages that come to and are issued from the daemon.

query Stops tracing queries that are handled by the daemon.

gsi Stops tracing the Group Services (internal) interface.

eval Stops tracing expression evaluation.

rdi Stops tracing the reliable daemon (internal) interface.

sched Stops tracing the internal scheduler.

shm Stops tracing shared memory management activity.

all Stops tracing all activities.

all_but_msgs
: Stops tracing all activities except for messages. Message activity is defined by the msgs argument.

Description

The haemtrcoff command is used to turn tracing off for specified activities of the Event Manager daemon. Trace output is placed in an Event Management trace log for the system partition.

Files

/var/ha/log/em.trace.cluster_name
Contains the trace log of the **haemd** daemon on the cluster named *cluster_name*

/var/ha/log/em.msgtrace.cluster_name
Contains message trace output from the Event Manager daemon on the cluster named *cluster_name*

Restrictions

Do not use this command during normal operation. Use this command only under the direction of the IBM Support Center. It provides information for debugging purposes and may degrade the performance of the event management subsystem or anything else that is running in the system partition.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/haemtrcoff Location of the **haemtrcoff** command

Examples

1. To turn off all tracing for the Event Management subsystem on one of the cluster nodes, log in to the node and enter:

   ```
   haemtrcoff -s emsvcs -a all
   ```

2. To turn off all tracing of initialization and configuration for the Event Management subsystem on a cluster node, log in to the node and enter:

   ```
   haemtrcoff -s emsvcs -a init,config
   ```

Related information

Commands: **haemtrcon**

Daemons: **haemd**

Scripts: **emsvcsctrl**

haemtrcon

Purpose

Turns tracing on for the event manager daemon.

Syntax

haemtrcon -s *subsys_name* **-a** *trace_list*

Flags

-s *cluster_name*
: Specifies the name of the event management subsystem. On a node, *cluster_name* is **emsvcs**. This flag and parameter must be specified.

-a *trace_list*
: Specifies a list of trace parameters. Each parameter specifies the type of activity for which tracing is to be turned on. At least one parameter must be specified. If more than one parameter is specified, the parameters must be separated by commas. The list cannot include blanks.

Parameters

The following trace parameters can be specified:

init Traces the initialization of the event manager daemon.

config Dumps information from the configuration file.

insts Traces resource variable instances that are handled by the daemon.

rmctrl Traces resource monitor control.

cci Traces the client communication (internal) interface.

emp Traces the event manager protocol.

obsv Traces resource variable observations.

evgn Traces event generation and notification.

reg Traces event registration and unregistration.

pci Traces the peer communication (internal) interface.

msgs Traces all messages that come to and are issued from the daemon.

query Traces queries that are handled by the daemon.

gsi Traces the group services (internal) interface.

eval Traces expression evaluation.

rdi Traces the reliable daemon (internal) interface.

sched Traces the internal scheduler.

shm Traces shared memory management activity.

all Traces all activities.

all_but_msgs
: Stops tracing all activities except for messages. Message activity is defined by the msgs argument.

regs Traces currently registered events.

dinsts Traces all resource variable instances known to the daemon.

iolists Traces immediate observation lists

olists Traces observation lists

Description

The **haemtrcon** command is used to turn tracing on for specified activities of the event manager daemon. Trace output is placed in an event management trace log for the system partition. When used, the **regs**, **dinsts**, **iolists**, and **olists** parameters perform a one-time trace. The specified information is placed in the trace log, but no further tracing is done.

Files

/var/ha/log/em.trace.*cluster_name*
Contains the trace log of the haemd daemon on the cluster named *cluster_name*.

/var/ha/log/em.msgtrace.cluster_name
Contains message trace output from the event manager daemon on the cluster named *cluster_name*.

Restrictions

Do not use this command during normal operation. Use this command only under the direction of the IBM Support Center. It provides information for debugging purposes and may degrade the performance of the event management subsystem or anything else that is running in the system partition.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/haemtrcon Location of the **haemtrcon** command

Examples

1. To turn on all tracing for the event management subsystem on one of the cluster nodes, log in to the node and enter:

   ```
   haemtrcon -s emsvcs -a all
   ```

2. To turn on all tracing of initialization and configuration for the event management subsystem on a cluster node, log in to the node and enter:

   ```
   haemtrcon -s emsvcs -a init,config
   ```

Related information

Commands: **haemtrcoff**

Daemons: **haemd**

Scripts: **emsvcsctrl**

haemunlkrm

Purpose

Unlocks and starts a resource monitor.

Syntax

haemunlkrm -s *subsys_name* **-a** *resmon_name*

Description

If the event management daemon cannot successfully start a resource monitor after three attempts within a two-hour interval, or if the daemon has successfully connected to the instances of a resource monitor *n* times within a two-hour interval, the resource monitor is "locked" and no further attempts are made to start it or to connect to any of its instances. *n* is **3** in an HACMP/ES cluster. Once the cause of the failure is determined and the problem corrected, the **haemunlkrm** command can be used to unlock the resource monitor and attempt to start it or connect to the resource monitor instances.

The status of the event manager daemon, as displayed by the **lssrc** command, indicates whether a resource monitor is locked.

Flags

-s *subsys_name*
: Specifies the name of the event management subsystem. On a node, *subsys_name* is **emsvcs**. This flag and parameter must be specified.

-a *resmon_name*
: Specifies the name of the resource monitor to unlock and start.

Implementation specifics

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

Location

/usr/sbin/rsct/bin/haemunlkrm
: Location of the **haemunlkrm** command

Examples

1. This example applies to unlocking a resource monitor on a node.

   If the output of the **lssrc** command indicates that the program resource monitor **IBM.PSSP.harmpd** is locked, correct the condition that prevented the resource monitor from being started and enter:

   ```
   haemunlkrm -s emsvcs -a IBM.PSSP.harmpd
   ```

Part 12. Appendixes

© Copyright IBM Corp. 2002, 2007

Appendix A. Accessibility features for RSCT

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in RSCT:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are discernible by touch but do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

The **IBM Cluster information center** and its related publications are accessibility-enabled. The accessibility features of the information center are described in the *Accessibility and keyboard shortcuts in the information center* topic at:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp

Related accessibility information

You can view these publications in Adobe Portable Document Format (PDF) using the Adobe Acrobat Reader. The PDFs are available from the **IBM Cluster information center**:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp

and the **IBM Publications Center**:

http://www.ibm.com/shop/publications/order

IBM and accessibility

See the **IBM Human Ability and Accessibility Center** for more information about the commitment that IBM has to accessibility:

http://www.ibm.com/able

© Copyright IBM Corp. 2002, 2007

Appendix B. Product-related information

Reliable Scalable Cluster Technology (RSCT) for AIX is a component of the following licensed programs:

- AIX 5L™ Versions 5.2 and 5.3
- AIX 6.1

RSCT version

This edition applies to:

- 2.3.12.0 (with mandatory APAR IZ06151) for AIX 5.2
- 2.4.8.0 (with mandatory APAR IZ06150) for AIX 5.3
- 2.5.0.0 (with mandatory APAR IZ06149) for AIX 6.1

To find out which version of RSCT is running on a particular AIX node, enter:

```
lslpp -L rsct.basic.rte
```

Product-related feedback

To contact the IBM cluster development organization, send your comments by e-mail to:

cluster@us.ibm.com

© Copyright IBM Corp. 2002, 2007

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

© Copyright IBM Corp. 2002, 2007

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department LRAS, Building 003
11400 Burnet Road
Austin, Texas 78758-3498
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

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.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which 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.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX
AIX 5L
eServer
HACMP
IBM
IBM (logo)
IBMLink
LoadLeveler
POWER
pSeries
RS/6000

SP
System p

Intel is a trademark of Intel Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be the trademarks or service marks of others.

Glossary

access control. The process of limiting access to system objects and resources to authorized principals.

access control list. A list of principals and the type of access allowed to each.

ACL. See *access control list.*

action. The part of the event response resource that contains a command and other information about the command.

attribute. Attributes are either persistent or dynamic. A resource class is defined by a set of persistent and dynamic attributes. A resource is also defined by a set of persistent and dynamic attributes. Persistent attributes define the configuration of the resource class and resource. Dynamic attributes define a state or a performance-related aspect of the resource class and resource. In the same resource class or resource, a given attribute name can be specified as either persistent or dynamic, but not both.

AIX. Advanced Interactive Executive. See *AIX operating system.*

AIX operating system. IBM's implementation of the UNIX operating system.

authentication. The process of validating the identity of an entity, generally based on user name and password. However, it does not address the access rights of that entity. Thus, it simply makes sure a user is who he or she claims to be.

authorization. The process of granting or denying access to an entity to system objects or resources, based on the entity's identity.

checksum. A count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived. If the counts match, it's assumed that the complete transmission was received. TCP and UDP communication layers provide a checksum count and verification as one of their services.

client. Client applications are the ordinary user interface programs that are invoked by users or routines provided by trusted services for other components to use. The client has no network identity of its own: it assumes the identity of the invoking user or of the process where it is called, who must have previously obtained network credentials.

cluster. A group of servers and other resources that act like a single system and enable high availability and, in some cases, load balancing and parallel processing.

clustering. The use of multiple computers (such as UNIX workstations, for example), multiple storage devices, and redundant interconnections to form what appears to users as a single highly-available system. Clustering can be used for load balancing, for high availability, and as a relatively low-cost form of parallel processing for scientific and other applications that lend themselves to parallel operations.

cluster security services. A component of RSCT that is used by RSCT applications and other RSCT components to perform authentication within both management domains and peer domains.

condition. A state of a resource as defined by the event response resource manager (ERRM) that is of interest to a client. It is defined by means of a logical expression called an event expression. Conditions apply to resource classes unless a specific resource is designated.

condition/response association. A link between a condition and a response.

CSM. Clusters Systems Management.

datagram. Synonymous with *UDP packet.*

DNS. See *domain name system.*

domain. (1) A set of network resources (such as applications and printers, for example) for a group of users. A user logs in to the domain to gain access to the resources, which could be located on a number of different servers in the network. (2) A group of server and client machines that exist in the same security structure. (3) A group of computers and devices on a network that are administered as a unit with common rules and procedures. Within the Internet, a domain is defined by its IP address. All devices that share a common part of the IP address are said to be in the same domain.

domain name. A meaningful and easy-to-remember ″handle″ for an Internet address.

domain name system. The service through which domain names are located and translated into IP addresses.

event. Occurs when the event expression of a condition evaluates to True. An evaluation occurs each time an instance of a dynamic attribute is observed.

event expression. A definition of the specific state when an event is true.

event response. One or more actions as defined by the event response resource manager (ERRM) that take place in response to an event or a rearm event.

© Copyright IBM Corp. 2002, 2007

failover. A backup operation that automatically switches to another adapter if one adapter fails. Failover is an important fault-tolerance function of mission-critical systems that rely on constant accessibility. Automatically and transparently to the user, failover redirects requests from the failed adapter to another adapter that mimics the operations of the failed adapter.

FFDC. See *first failure data capture.*

first failure data capture. Provides a way to track problems back to their origin even though the source problem may have occurred in other layers or subsystems than the layer or subsystem with which the end user is interacting. FFDC provides a correlator called an **ffdc_id** for any error that it writes to the AIX error log. This correlator can be used to link related events together to form a chain.

FIFO. First in first out, usually referring to buffers.

High Performance Switch. The switch that works in conjunction with a specific family of IBM System p servers.

HPS. See *High Performance Switch.*

Internet Protocol. The method by which data is sent from one computer to another on the Internet.

IP. See *Internet Protocol.*

IP address. A 32-bit (in IP Version 4) or 128-bit (in IP Version 6) number identifying each sender or receiver of information that is sent in packets across the Internet.

LAPI. See *low-level application programming interface.*

LAPI for Linux. The version of LAPI that runs on the Linux operating system. See also *low-level application programming interface.*

Linux. A freeware clone of UNIX for 386-based personal computers (PCs). Linux consists of the **linux** kernel (core operating system), originally written by Linus Torvalds, along with utility programs developed by the Free Software Foundation and by others.

LoadLeveler®. A job management system that works with POE to let users run jobs and match processing needs with system resources, in order to make better use of the system.

low-level application programming interface. A low-overhead message-passing protocol that uses a one-sided communication model and active message paradigm to transfer data among tasks. See also *LAPI for Linux* and *RSCT LAPI for AIX.* Contrast with *PSSP LAPI.*

logical unit number. A unique identifier used on a SCSI bus that enables it to differentiate between up to eight separate devices (each of which is a logical unit). Each LUN is a unique number that identifies a specific logical unit, which may be an end user, a file, or an application program.

LUN. See *logical unit number.*

management domain. A set of nodes configured for manageability by the Clusters Systems Management (CSM) licensed program. Such a domain has a management server that is used to administer a number of managed nodes. Only management servers have knowledge of the whole domain. Managed nodes only know about the servers managing them; they know nothing of each other. Contrast with *peer domain.*

Message Passing Interface. A standardized API for implementing the message-passing model.

MPI. See *Message Passing Interface.*

mutex. See *mutual exclusion object.*

mutual exclusion object. A program object that allows multiple program threads to share the same resource, such as file access, but not simultaneously. When a program is started, a mutual exclusion object is created with a unique name. After this stage, any thread that needs the resource must lock the mutual exclusion object from other threads while it is using the resource. The mutual exclusion object is set to unlock when the data is no longer needed or the routine is finished.

network credentials. These represent the data specific to each underlying security mechanism.

OSI. Operating system image.

PAC. See *privileged attribute certificate.*

packet. The unit of data that is routed between an origin and a destination on the Internet or any other packet-switched network.

Parallel Environment. An IBM licensed program that is an execution and development environment for parallel C, C++, and FORTRAN programs. PE also includes tools for debugging, profiling, and tuning parallel programs.

parallel operating environment. An execution environment that smooths the differences between serial and parallel execution. It lets you submit and manage parallel jobs.

Parallel System Support Programs. The IBM Parallel System Support Programs for AIX 5L licensed program is system administration software for the IBM RS/6000 SP system.

PE. See *Parallel Environment.*

peer domain. A set of nodes configured for high availability by the configuration resource manager. Such a domain has no distinguished or master node. All

nodes are aware of all other nodes, and administrative commands can be issued from any node in the domain. All nodes also have a consistent view of the domain membership. Contrast with *management domain.*

POE. See *parallel operating environment.*

port. A ″logical connection place″. Using TCP/IP, the way a client program specifies a particular server program on a computer in a network.

principal. A user, an instance of the server, or an instance of a trusted client whose identity is to be authenticated.

privileged attribute certificate. Contains such information as the client's name and the groups to which it belongs. Its format is dependent on the underlying security mechanism.

protocol. The set of rules that endpoints in a telecommunication connection use when they communicate.

PSSP. See *Parallel System Support Programs.*

PSSP LAPI. The version of LAPI that supports the SP Switch2.

rearm event. Occurs when the rearm expression for a condition evaluates to True.

rearm expression. An expression that generates an event which alternates with an original event in the following way: the event expression is used until it is true; then, the rearm expression is used until it is true; then, the event expression is used. The rearm expression is commonly the inverse of the event expression. It can also be used with the event expression to define an upper and lower boundary for a condition of interest.

Reliable Scalable Cluster Technology. A set of software components that together provide a comprehensive clustering environment for AIX, Linux, and Windows. RSCT is the infrastructure used by a variety of IBM products to provide clusters with improved system availability, scalability, and ease of use.

resource. An entity in the system that provides a set of services. Examples of hardware entities are processors, disk drives, memory, and adapters. Examples of software entities are database applications, processes, and file systems. Each resource in the system has one or more attributes that define the state of the resource.

resource class. A broad category of system resource, for example: node, file system, adapter. Each resource class has a container that holds the functions, information, dynamic attributes, and conditions that apply to that resource class. For example, the **/tmp space used** condition applies to a file system resource class.

resource manager. A process that maps resource and resource-class abstractions into calls and commands for one or more specific types of resources. A resource manager can be a standalone daemon, or it can be integrated into an application or a subsystem directly.

RSCT. See *Reliable Scalable Cluster Technology.*

RSCT LAPI for AIX. The version of LAPI that runs on the AIX 5.2, 5.3, and 6.1 operating systems. See also *low-level application programming interface.*

RSCT peer domain. See *peer domain.*

SCSI. See *Small System Computer Interface.*

Small System Computer Interface. A parallel interface that can have up to eight devices all attached through a single cable; the cable and the host (computer) adapter make up the SCSI bus. The bus allows the interchange of information between devices independently of the host. In the SCSI program, each device is assigned a unique number, which is either a number between 0 and 7 for an 8-bit (narrow) bus, or between 8 and 16 for a 16-bit (wide) bus. The devices that request input/output (I/O) operations are initiators and the devices that perform these operations are targets. Each target has the capacity to connect up to eight additional devices through its own controller; these devices are the logical units, each of which is assigned a unique number for identification to the SCSI controller for command processing.

SD. Structured data.

security context token. A pointer to an opaque data structure called the context token descriptor. The context token is associated with a connection between a client and the server.

security services token. A pointer to an opaque descriptor called the security token descriptor. It keeps track of the mechanism-independent information and state.

servers. Server programs are usually daemons or other applications running in the background without a user's inherited credentials. A server must acquire its own network identity to get to access other trusted services.

SP Switch2. The switch that works in conjunction with IBM RS/6000 SP systems.

standalone system. A system on which you are using LAPI for Linux or RSCT LAPI for AIX that is *not* running PE.

striping. The distribution of message data across multiple communication adapters in order to increase bandwidth.

TCP. See *Transmission Control Protocol.*

Transmission Control Protocol. One of the core Internet protocols. TCP ports are 16-bit entities, so a maximum of 65535 different endpoints are possible within a single IP address.

UDP. See *User Datagram Protocol.*

User Datagram Protocol. One of the core Internet protocols. UDP is a layer 4 protocol (Transport layer of the OSI model) within the Internet protocol suite. It provides a mechanism to identify different endpoints on a single host by using ports. UDP deals with single-packet delivery that is provided by the underlying IP. As a stateless protocol, it is often used in applications where data must arrive quickly. This smaller feature set provides quicker data transmittal and lower total overhead. UDP packets (or *datagrams*) contain, in addition to the lower-level headers, a UDP header, which consists of the packet length, source and destination ports, and a checksum. UDP ports are 16-bit entities, so a maximum of 65535 different endpoints are possible within a single IP address.

Index

A

access control lists (ACLs)
  least-privilege (LP) 377
accessibility features 615
ACLs
  least-privilege (LP) 377
addrpnode command 84
audience of this book ix

B

bibliography x
books
  RSCT x

C

cfgvsd command 474
chcomg command 88
chcondition command 188
chlpclacl command 378
chlpcmd command 350
chlpracl command 383
chlpriacl command 389
chlprsacl command 394
chresponse command 193
chrmcacl command 4
chrsrc command 14
chsensor command 266
CIM resource manager
  commands 175
command-line interface
  See commands
commands
  addrpnode 84
  cfgvsd 474
  chcomg 88
  chcondition 188
  chlpclacl 378
  chlpcmd 350
  chlpracl 383
  chlpriacl 389
  chlprsacl 394
  chresponse 193
  chrmcacl 4
  chrsrc 14
  chsensor 266
  Common Information Model (CIM) resource manager 175
  configuration resource manager 83
  createvsd 476
  ctaclfck 306
  ctadmingroup 146
  cthactrl 434
  cthagsctrl 454
  cthagstune 458
  cthatsctrl 440
commands *(continued)*
  cthatstune 443
  ctlvsd 482
  ctmsskf 311
  ctscachgen 315
  ctscfg 318
  ctsidmck 322
  ctskeygen 326
  ctsnap 544
  ctsthl 329
  ctsvhbac 335
  ctsvhbal 339
  ctsvhbar 342
  defvsd 485
  elogevent 256
  event-response resource manager (ERRM) 187
    elogevent 256
    logevent 256
    lsevent 210
  fccheck 548
  fcclear 550
  fcdecode 554
  fcdispfid 556
  fcfilter 558
  fcinit 560
  fclogerr 565
  fcpushstk 574
  fcreport 582
  fcstkrpt 585
  fcteststk 588
  fencevsd 487
  forcerpoffline 92
  grpsvcsctrl 460
  ha_vsd 488
  ha.vsd 489
  haemqvar 604
  haemtrcoff 608
  haemtrcon 610
  haemunlkrm 612
  hagsns 465
  hagsvote 467
  hatsoptions 446
  least-privilege (LP) access control lists (ACLs) 377
  least-privilege (LP) resource manager 349
  logevent 256
  LP ACLs 377
  lphistory 354
  lsactdef 19
  lsassocmap 176
  lsaudrec 292
  lscomg 94
  lscondition 198
  lscondresp 204
  lsevent 210
  lslpclacl 399
  lslpcmd 359
  lslpracl 404
  lslpriacl 410

© Copyright IBM Corp. 2002, 2007

commands *(continued)*
lslprsacl 415
lsresponse 215
lsrpdomain 98
lsrpnode 101
lsrsrc 24
lsrsrcassoc 178
lsrsrcdef 31
lssensor 270
lsvsd 493
mkcimreg 181
mkcomg 106
mkcondition 221
mkcondresp 228
mklpcmd 364
mkresponse 231
mkrpdomain 110
mkrsrc 38
mksensor 276
nlssrc 436
preparevsd 496
preprpnode 117
refrsrc 43
refsensor 282
removevsd 498
resetrsrc 45
resource monitoring and control (RMC) 13
resumevsd 500
rmaudrec 298
rmcctrl 8
rmcomg 120
rmcondition 237
rmcondresp 240
rmlpcmd 369
rmresponse 244
rmrpdomain 123
rmrpnode 126
rmrsrc 50
rmsensor 286
runact 54
runlpcmd 372
rvsdrestrict 502
sensor resource manager 265
snmpevent 262
startcondresp 247
startrpdomain 129
startrpnode 133
startrsrc 59
startvsd 504
statvsd 506
stopcondresp 251
stoprpdomain 136
stoprpnode 139
stoprsrc 64
stopvsd 508
suspendvsd 510
ucfgvsd 512
undefvsd 514
unfencevsd 515
updatevsdnode 517
updatevsdtab 520
commands *(continued)*
updatevsdvg 521
vsdatalst 524
vsdchgserver 526
vsdelnode 528
vsdelvg 530
vsdnode 532
vsdsklst 535
vsdvg 537
vsdvgts 539
Common Information Model (CIM)
commands 175
configuration resource manager commands 83
control scripts
grpsvcsctrl 460
topsvcsctrl 449
conventions
terminology x
typographic ix
createvsd command 476
ct_class_ids file 152
ct_cssk.kf file 153
ct_has.pkf file 154
ct_has.qkf file 156
ct_has.thl file 158
ctaclfck command 306
ctadmingroup command 146
ctcas_hba2.map file 150
ctcasd daemon 309
ctcasd.cfg file 160
cthactrl command 434
cthagsctrl command 454
cthagstune command 458
cthatsctrl command 440
cthatstune command 443
ctlvsd command 482
ctmsskf command 311
ctrmc.acls file 165
ctscachgen command 315
ctscfg command 318
ctsec_map.global file 167
ctsec_map.local file 167
ctsec.cfg file 166
ctsidmck command 322
ctskeygen command 326
ctsnap command 544
ctsthl command 329
ctstrtcasd utility 333
ctsvhbac command 335
ctsvhbal command 339
ctsvhbar command 342

D

daemons
ctcasd 309
haemd 602
hagsd 464
definitions 623
defvsd command 485

E

elogevent script 256
emsvcsctrl script 598
enotifyevent script 258
ERRM
  event information
    logging 256
ERRM commands
  elogevent 256
  logevent 256
  lsevent 210
  snmpevent 262
ERRM scripts
  elogevent 256
  logevent 256
  snmpevent 262
event information
  logging 256
event-response resource manager (ERRM)
  commands 187
    elogevent 256
    logevent 256
    lsevent 210
  event information
    logging 256
  scripts
    elogevent 256
    logevent 256
ewallevent script 260

F

fccheck command 548
fcclear command 550
fcdecode command 554
fcdispfid command 556
fcfilter command 558
fcinit command 560
fclogerr command 565
fcpushstk command 574
fcreport command 582
fcstkrpt command 585
fcteststk command 588
feedback
  product-related 617
fencevsd command 487
files
  ct_class_ids 152
  ct_cssk.kf 153
  ct_has.pkf 154
  ct_has.qkf 156
  ct_has.thl 158
  ctcas_hba2.map 150
  ctcasd.cfg 160
  ctrmc.acls 165
  ctsec_map.global 167
  ctsec.cfg 166
  unix.map 172
forcerpoffline command 92

G

glossary 623
group services
  control commands
    cthagsctrl 454
  tuning 458
grpsvcsctrl command 460

H

ha_vsd command 488
ha.vsd command 489
haemd daemon 602
haemd_HACMP program 603
haemqvar command 604
haemtrcoff command 608
haemtrcon command 610
haemunlkrm command 612
hagsd daemon 464
hagsns command 465
hagsvote command 467
hatsoptions command 446

I

interface, command-line
  See commands

L

least-privilege (LP)
  access control lists (ACLs) 377
  man pages 421
least-privilege resource manager
  commands 349
logevent script 256
LookAt xii
LP
  access control lists (ACLs) 377
  man pages 421
LP resource manager
  commands 349
lpacl man page 422
lphistory command 354
lsactdef command 19
lsassocmap command 176
lsaudrec command 292
lscomg command 94
lscondition command 198
lscondresp command 204
lsevent command 210
lslpclacl command 399
lslpcmd command 359
lslpracl command 404
lslpriacl command 410
lslprsacl command 415
lsresponse command 215
lsrpdomain command 98
lsrpnode command 101
lsrsrc command 24

lsrsrcassoc command 178
lsrsrcdef command 31
lssensor command 270
lsvsd command 493

M

man pages
  least-privilege (LP) 421
  LP 421
  lpacl 422
  resource_data_input 70
  RMC 69
  rmccli 74
mkcimreg command 181
mkcomg command 106
mkcondition command 221
mkcondresp command 228
mklpcmd command 364
mkresponse command 231
mkrpdomain command 110
mkrsrc command 38
mksensor command 276

N

nlssrc command 436
notifyevent script 258

P

preparevsd command 496
preprpnode command 117
prerequisite information x
prerequisite knowledge for this book ix
product-related feedback 617
programs
  haemd_HACMP 603
publications
  RSCT x

R

refrsrc command 43
refsensor command 282
related information x
Reliable Scalable Cluster Technology (RSCT)
  commands
    cthactrl 434
Reliable Scalable Cluster Technology (RSCT) topology services
  control scripts
    topsvcsctrl 449
  scripts
    topsvcs 448
removevsd command 498
resetrsrc command 45
resource manager commands
  least-privilege (LP) 349
resource managers
  Common Information Model(CIM)
    commands 175
  sensor
    commands 265
resource monitoring and control
  commands 13
resource_data_input man page 70
resumevsd command 500
rmaudrec command 298
RMC
  commands 13
  man pages 69
rmccli man page 74
rmcctrl command 8
rmcomg command 120
rmcondition command 237
rmcondresp command 240
rmlpcmd command 369
rmresponse command 244
rmrpdomain command 123
rmrpnode command 126
rmrsrc command 50
rmsensor command 286
RSCT
  books x
  feedback 617
  publications x
  version 617
RSCT topology services
  control scripts
    topsvcsctrl 449
  scripts
    topsvcs 448
runact command 54
runlpcmd command 372
rvsdrestrict command 502

S

scripts
  elogevent 256
  emsvcsctrl 598
  enotifyevent 258
  event-response resource manager (ERRM)
    elogevent 256
    logevent 256
  ewallevent 260
  grpsvcsctrl 460
  logevent 256
  notifyevent 258
  snmpevent 262
  topsvcs 448
  topsvcsctrl 449
  wallevent 260
sensor resource manager
  commands 265
snmpevent script 262
startcondresp command 247
startrpdomain command 129
startrpnode command 133

startrsrc command 59
startvsd command 504
statvsd command 506
stopcondresp command 251
stoprpdomain command 136
stoprpnode command 139
stoprsrc command 64
stopvsd command 508
subsystem
  control commands
    cthactrl 434
    cthagsctrl 454
    cthatsctrl 440
  group services
    tuning 458
  topology services
    tuning 443
suspendvsd command 510

T

terminology 623
terminology conventions x
topology services
  control commands
    cthatsctrl 440
  introduction 439
  tuning 443
topology services subsystem
  control scripts
    topsvcsctrl 449
  scripts
    topsvcs 448
topsvcs script 448
topsvcsctrl script 449
trademarks 620
tuning
  group services 458
  topology services 443
typographic conventions ix

U

ucfgvsd command 512
undefvsd command 514
unfencevsd command 515
unix.map file 172
updatevsdnode command 517
updatevsdtab command 520
updatevsdvg command 521
utilities
  ctstrtcasd 333

V

version
  of RSCT 617
vsdatalst command 524
vsdchgserver command 526
vsdelnode command 528
vsdelvg command 530
vsdnode command 532
vsdsklst command 535
vsdvg command 537
vsdvgts command 539

W

wallevent script 260

Readers' comments – We'd like to hear from you

Reliable Scalable Cluster Technology for AIX
Technical Reference

Publication No. SA22-7890-14

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.

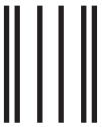
If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	E-mail address

Readers' Comments — We'd Like to Hear from You
SA22-7890-14

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department H6MA, Mail Station P181
2455 South Road
Poughkeepsie, NY 12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5765-E62, 5765-G03, 5765-G62

SA22-7890-14