Reliable Scalable Cluster Technology
Version 3.1.5.0

Technical Reference: RSCT for Multiplatforms

Reliable Scalable Cluster Technology
Version 3.1.5.0

Technical Reference: RSCT for Multiplatforms

Note
Before using this information and the product it supports, read the information in "Notices" on page 399.

This edition applies to Reliable Scalable Cluster Technology Version 3.1.5.0 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this document

This publication describes Reliable Scalable Cluster Technology (RSCT) commands, daemons, files, man pages, scripts, and utilities for Linux, Solaris, and Windows.

Highlighting

The following highlighting conventions are used in this document:

Table 1. Conventions

Convention	Usage
bold	**Bold** words or characters represent system elements that you must use literally, such as commands, flags, path names, directories, file names, values, PE component names (**poe**, for example), and selected menu options.
bold underlined	**Bold underlined** keywords are defaults. These take effect if you do not specify a different keyword.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface.
italic	*Italic* words or characters represent variable values that you must supply. *Italics* are also used for information unit titles, for the first use of a glossary term, and for general emphasis in text.
<*key*>	Angle brackets (less-than and greater-than) enclose the name of a key on the keyboard. For example, <**Enter**> refers to the key on your terminal or workstation that is labeled with the word *Enter*.

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

{*item*}	Braces enclose a list from which you must choose an item in format and syntax descriptions.
[*item*]	Brackets enclose optional items in format and syntax descriptions.
<**Ctrl-***x*>	The notation <**Ctrl-***x*> indicates a control character sequence. For example, <**Ctrl-c**> means that you hold down the control key while pressing <**c**>.
item...	Ellipses indicate that you can repeat the preceding item one or more times.
|	• In *synopsis* or *syntax* statements, vertical lines separate a list of choices. In other words, a vertical line means *Or*. • In the left margin of the document, vertical lines indicate technical changes to the information.

Entering commands

When you work with the operating system, you typically enter commands following the shell prompt on the command line. The shell prompt can vary. In the following examples, $ is the prompt.

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

```
$ ls
```

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

The general format for entering operating system commands is:

Command Flag(s) Parameter

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

```
$ ls -l
```

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

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

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

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

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

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

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

Case sensitivity in AIX

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

ISO 9000

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

RSCT versions

This edition applies to RSCT version, release, modification, and fix number 3.1.5.0.

You can use the **ctversion** command to find out which version of RSCT is running on a particular AIX, Linux, Solaris, or Windows node. For example:

```
/usr/sbin/rsct/install/bin/ctversion
```

The following output is displayed:

```
# /usr/sbin/rsct/install/bin/ctversion
rlis1313a 3.1.5.0
```

where, `rlis1313a` is the RSCT build level.

On the AIX operating system, you can also use the **lslpp** command to find out which version of RSCT is running on a particular AIX node. For example:

```
lslpp -L rsct.core.utils
```

The following output is displayed:

```
  Fileset                      Level    State  Type  Description (Uninstaller)
  ----------------------------------------------------------------------------
  rsct.core.utils              3.1.5.0    C     F    RSCT Utilities

State codes:
 A -- Applied.
 B -- Broken.
 C -- Committed.
 E -- EFIX Locked.
 O -- Obsolete.  (partially migrated to newer version)
 ? -- Inconsistent State...Run lppchk -v.

Type codes:
 F -- Installp Fileset
 P -- Product
 C -- Component
 T -- Feature
 R -- RPM Package
```

On the Linux operating system, you can also use the **rpm** command to find out which version of RSCT is running on a particular Linux or Solaris node. For example:

```
rpm -qa | grep rsct.basic
```

On the Windows operating system, you can also perform the following steps to find out which version of RSCT is running on a particular Windows node:

1. Click the Windows **start** button.
2. Select **All Programs**.
3. Select **Tivoli SA MP Base**.
4. Click **SA MP Version**.

Related information

The following PDF documents that contain RSCT information can be found at Reliable Scalable Cluster Technology (RSCT) PDFs:

- *Administering RSCT*
- *Messages for RSCT*
- *Programming Group Services for RSCT*
- *Programming RMC for RSCT*
- *Technical Reference: RSCT for AIX*
- *Troubleshooting RSCT*

Technical Reference: RSCT for Multiplatforms

This publication is intended for system administrators who want to use RSCT commands, daemons, files, scripts, and utilities on Linux, Solaris, Windows, or a combination of these operating systems. The system administrator should be experienced with UNIX and networked systems.

What's new in Technical Reference: RSCT for Multiplatforms

Read about new or significantly changed information for the Technical Reference: RSCT for Multiplatforms topic collection.

How to see what's new or changed

In this PDF file, you might see revision bars (|) in the left margin that identifies new and changed information.

June 2014

- Added information about the **-s** and **-w** flags to the "startrpdomain Command" on page 105, "startrpnode Command" on page 108, "stoprpdomain Command" on page 110, and "stoprpnode Command" on page 112 commands.

November 2013

The following information is a summary of the updates made to this topic collection:

- Added information about the *node_name@host_name* parameter to the "addrpnode Command" on page 65.
- Added information about the **-p** flag to the "mkrpdomain Command" on page 88.
- Added information about the **-b** and **-B** flags to the "rmcctrl Command" on page 7.

The resource monitoring and control (RMC) subsystem

The resource monitoring and control (RMC) subsystem is the scalable backbone of RSCT that provides a generalized framework for managing and monitoring resources (physical or logical system entities) within a single system or a cluster.

RMC control commands

cfgrmcsnmp
Purpose

Controls the ability of RMC to receive traps on Linux hosts.

Syntax

cfgrmcsnmp [**-u**] [**-h**]

Description

The **cfgrmcsnmp** command sets up or uninstalls RMC's ability to receive traps on Linux hosts. This command checks the appropriate daemons and either creates the **/usr/share/snmp/snmptrapd.conf** file or appends to it.

The **trap2rmc** command is configured when the **cfgrmcsnmp** command has been run. The **snmptrapd.conf** file is installed or appended to in the **/usr/share/snmp/** directory. This configuration file contains entries to call **trap2rmc** when traps are received. The installation script also restarts the **snmptrapd** daemon, so once the installation is complete, trap logging will be done automatically without the user needing to set anything up. However, if the **/usr/share/snmp/snmptrapd.conf** file exists, it will be saved as a backup file (**/usr/share/snmp/snmptrapd.conf.orig**) and the administrator will need to configure the two configuration files manually to get the desired results. If the UCD-SNMP daemon (**snmptrapd**) has been removed before installation begins, the installation will not try to install the UCD-SNMP package.

Flags

-u Uninstalls RMC's ability to receive traps in the audit log.

-h Writes this 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.

Files

/usr/share/snmp/snmptrapd.conf
Contains entries to call **trap2rmc** when traps are received

/usr/share/snmp/snmptrapd.conf.orig
Backup file for **/usr/share/snmp/snmptrapd.conf**

Implementation specifics

This command is part of the **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/install/bin/cfgrmcsnmp

Related reference:

"trap2rmc" on page 10

chrmcacl Command
Purpose

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

Syntax

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

Description

This command is used 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 *Administering RSCT* 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 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 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 must have execute (**x**) access to this command.

Implementation specifics

This command is part of the **rsct.core** fileset for AIX and **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

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 is:

```
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 is:

```
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 is:

```
IBM.Sensor

    joe@Host1.CoX.com   C       rw

        Host1.CoX.com   *       r
```

   upon successful completion of the command.

recfgct Command

Purpose

Reconfigures the Reliable Scalable Cluster Technology (RSCT) subsystems.

Syntax

/usr/sbin/rsct/install/bin/recfgct [**-n** | **-s** | **-h**]

Description

Attention: Use this command with extreme caution.

The **recfgct** command is used to remove all RSCT data under the **/var/ct** directory, generate a new node ID, and make it appear as if the RSCT components are just installed. Because of the destructive nature of

this command, it is not normally started by the system administrator. You must use this command *only* if you need to remove a duplicate node ID or if an IBM® service representative instructs you to use it.

When RSCT is first installed, a node ID is automatically generated. The node ID is a true random 64-bit number. Each system where RSCT is installed must have a unique node ID. If a copy of an operating system image (OSI) that has RSCT installed on it is installed on another system, the other system has the same node ID as the system from which the copy is made. This is referred to as *cloning*. For AIX platform, cloning is typically performed using such AIX-supported commands and procedures as **mksysb**. These commands and procedures call **recfgct** automatically. For other platforms, the **recfgct** command must be run immediately after a cloned OSI is installed.

If the **-s** flag is specified, after all data under the **/var/ct** directory is removed, the node ID contained in the **/etc/ct_node_id** file is used to re-create the **/var/ct/cfg/ct_node_id** file.

The **-h** flag is supported on RSCT 2.4.9.1 (or later) for AIX 5.3, on RSCT 2.5.1.1 (or later) for AIX 6.1 and on RSCT 3.1.0.0 for AIX 7.1 for AIX platform and RSCT 2.5.1.1 (or later) for other platforms. If you try to run the **recfgct -h** command on a prior version of RSCT, the **-h** flag is ignored and all RSCT data is removed.

Flags

-n Generate a new node ID. It is the default behavior if no option is specified.

-s Saves the node ID.

-h Writes the command usage statement to standard output and then exits.

The **-h** flag is supported on RSCT 2.4.9.1 (or later) for AIX 5.3, on RSCT 2.5.1.1 (or later) for AIX 6.1 and on RSCT 3.1.0.0 for AIX 7.1 for AIX platform and RSCT 2.5.1.1 (or later) for other platforms. If you try to run the **recfgct -h** command on a prior version of RSCT, the **-h** flag is ignored and all RSCT data is removed.

Restrictions

The **-h** flag is supported on RSCT 2.4.9.1 (or later) for AIX 5.3, on RSCT 2.5.1.1 (or later) for AIX 6.1 and on RSCT 3.1.0.0 for AIX 7.1 and RSCT 2.5.1.1 (or later) for other platforms. If you try to run the **recfgct -h** command on a prior version of RSCT, the **-h** flag is ignored and all RSCT data is removed.

Files

/etc/ct_node_id
Contains a copy of the RSCT node ID

/var/ct/cfg/ct_node_id
Contains the RSCT node ID

Standard output

When the **-h** flag is specified, this command usage statement is written to standard output and then the command exits.

The **-h** flag is supported on RSCT 2.4.9.1 (or later) for AIX 5.3, on RSCT 2.5.1.1 (or later) for AIX 6.1 and on RSCT 3.1.0.0 for AIX 7.1 and RSCT 2.5.1.1 (or later) for other platforms. If you try to run the **recfgct -h** command on a prior version of RSCT, the **-h** flag is ignored and all RSCT data is removed.

Exit status

0 The command ran successfully.

1 The command did not run successfully.

Security

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

Implementation specifics

This command is part of the **rsct.core** fileset for the AIX operating system and **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows operating system, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/install/bin/recfgct

Examples

1. After installing a cloned operating system image, enter:
   ```
   /usr/sbin/rsct/install/bin/recfgct
   ```

rmcctrl Command
Purpose

Manages the resource monitoring and control (RMC) subsystem.

Syntax

rmcctrl { **-a** | **-A** | **-b** || **-B** || **-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 started.

Note: While the RMC subsystem can be stopped and started by using the **stopsrc** and **startsrc** commands, you can use the **rmcctrl** command 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.

-b Sets the idle timeout for the RMC API client session to n seconds. If the RMC daemon finds no activity in the session for the last n seconds, it is closed.

-B Sets the idle timeout for the RMC API client session to a default value of 0 seconds (that is it is disabled).

-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 that is listed in the *RSCT: 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 by 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 by using the User Datagram Protocol (UDP).

This flag is supported on RSCT release 2.4.1.0 or later. 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 can 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. This timeout value must include the following actions:

- Receiving the first message of the start session protocol after the RMC subsystem accepts a client connection.
- Receiving the complete client message by the RMC subsystem, after the initial message is 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 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 stopped.

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

Security

Privilege control: only the root user must run (**x**) access to this command.

Exit Status

0 The command is successful.

1 The command was not successful.

Standard Output

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

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
   ```

Location

/usr/sbin/rsct/bin/rmcctrl

trap2rmc
Purpose

Formats a trap received from the SNMP manager into a user-readable message and enters it in the IBM.AuditLog.

Syntax

trap2rmc

Description

The **trap2rmc** command is configured when the **cfgrmcsnmp** command has been run. A new **snmptrapd.conf** file is installed in the **/usr/share/snmp/** directory. This configuration file contains entries to call **trap2rmc** when traps are received. The installation script also restarts the **snmptrapd** daemon, so once the installation is complete, trap logging will be done automatically without the user needing to set anything up. However, if the **/usr/share/snmp/snmptrapd.conf** file exists, it will be saved as a backup file (**/usr/share/snmp/snmptrapd.conf.orig**) and the administrator will need to configure the two configuration files manually to get the desired results. If the UCD-SNMP daemon (**snmptrapd**) has been removed before installation begins, the installation will not try to install the UCD-SNMP package. Instead, it will still install the **/usr/share/snmp/snmptrapd.conf** file, but it will have no effect.

Files

/usr/share/snmp/snmptrapd.conf

Implementation specifics

This command is part of the **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/trap2rmc

Related reference:

"cfgrmcsnmp" on page 1

RMC commands

chrsrc Command
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.*

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.

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"
```

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.

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.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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 *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}]'
   ```

Location

/usr/sbin/rsct/bin/chrsrc

lsactdef Command
Purpose

Displays the action definitions of a resource or a resource class.

Syntax

To display the action definitions of a *resource:*

lsactdef [**-p** *property*] [**-s i** | **o**] [**-e**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*action1* [*action2 ...*]]

To display the action definitions of a *resource class:*

lsactdef -c [**-p** *property*] [**-s i** | **o**] [**-e**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*action1* [*action2 ...*]]

To display all resource class names:

lsactdef

Description

The **lsactdef** command displays a list of the action definitions of a resource or a resource class. By default, this command displays the action definitions of a *resource.* To see the action definitions of a *resource class,* specify the **-c** flag.

If you do not specify any actions on the command line, this command only displays actions that are defined as **public**. To override this default, use the **-p** flag or specify on the command line the names of the actions that have definitions you want to display.

To see the structured data definition that is required as input when this action is invoked, specify the **-s i** flag. To see the structured data definition linked with the output that results from invoking this action, specify the **-s o** flag.

By default, this command does not display action descriptions. To display action definitions and descriptions, specify the **-e** flag.

Flags

-c Displays the action definitions for *resource_class*.

-d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify a delimiter other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e Specifies expanded format. Displays descriptions along with the action definitions.

-i Specifies input format. Generates a template of *resource_data_input_file*. The output is displayed in long (stanza) format. The attribute's SD element data types are displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the **lsactdef** command be directed to a file. This flag overrides the **-s o** flag.

-l Specifies "long" format — one entry per line. This is the default display format. If the **lsactdef** command is issued with the **-l** flag, but without a resource class name, the **-l** flag is ignored when the command returns the list of defined resource class names.

-p *property*
Displays actions with the specified *property*. By default, only the definitions for public actions are displayed. To display all action definitions regardless of the action property, use the **-p 0** flag.

Action properties:

0x0001 long_running

0x0002 public

A decimal or hexadecimal value can be specified for the property. To request the action definitions for all actions that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the **-p** flag. For example, to request the action definitions for all actions that are **long_running** or **public**, enter:

```
-p 0x03
```

-s i | o
Displays the structured data definition for the action input or action response.

i Displays the action input structured data definitions. This is the default.

o Displays the action response (output) structured data definitions.

-t Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x Suppresses header printing.

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

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

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

Parameters

resource_class
Specifies the name of the resource class with the action definitions that you want to display. If *resource_class* is not specified, a list of all of the resource class names is displayed.

action1 **[***action2...***]**
Specifies one or more actions. If *resource_class* is specified, zero or more action names can be specified. If no actions are specified, all of the action definitions for *resource_class* are displayed. Enter specific action names to control which actions are displayed and in what order. Use blank spaces to separate action names.

Security

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

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system

where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all of the resource classes, enter:

   ```
   lsactdef
   ```

   The output will look like this:

   ```
   class_name
   "IBM.Association"
   "IBM.AuditLog"
   "IBM.AuditLogTemplate"
   "IBM.Condition"
   "IBM.EventResponse"
   "IBM.Host"
   "IBM.Program"
   "IBM.Sensor"
   "IBM.ManagedNode"
   ...
   ```

2. To list the public resource action definitions for resource class IBM.AuditLog, enter:

   ```
   lsactdef IBM.AuditLog
   ```

   The output will look like this:

```
Resource Action Definitions for
class_name: IBM.AuditLog
action 1:
        action_name    = "GetRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 0
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
action 2:
        action_name    = "DeleteRecords"
        display_name   = ""
        description    = ""
        properties     = {"public"}
        confirm_prompt = ""
        action_id      = 1
        variety_list   = {{1..1}}
        variety_count  = 1
        timeout        = 0
....
```

3. To list the structured data definition required for invoking the action on resources in resource class IBM.AuditLog, action GetRecords, enter:

```
lsactdef -s i IBM.AuditLog GetRecords
```

The output will look like this:

```
Resource Action Input for: IBM.AuditLog
action_name GetRecords:
sd_element 1:
        element_name      = "MatchCriteria"
 display_name      = ""
 description       = ""
        element_data_type = "char_ptr"
 element_index     = 0
sd_element 2:
        element_name      = "IncludeDetail"
 display_name      = ""
 description       = ""
        element_data_type = "uint32"
        element_index     = 1
```

Location

/usr/sbin/rsct/bin/lsactdef

lsrsrc Command

Purpose

Displays attributes and values for a resource or a resource class.

Syntax

To display the attributes and values for a *resource:*

lsrsrc [**-s** *"selection_string"*] [**-a** | **-N** { *node_file* | **"-"** }] [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*resource_class*] [*attr...*]

lsrsrc -r [**-s** *"selection_string"*] [**-a** | **-N** { *node_file* | **"-"** }] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*resource_class*]

To display the attributes and values for a *resource class:*

lsrsrc -c [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-a**] [**-h**] [**-TV**] *resource_class* [*attr*...]

lsrsrc -C *domain_name...* [**-A p** | **d** | **b**] [**-p** *property*] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display a list of all of the resource classes:

lsrsrc

Description

The **lsrsrc** command displays the persistent and dynamic attributes and their values for a resource or a resource class.

Instead of specifying multiple node names in *selection_string*, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "-"** to read the node names from standard input.

When one or more attribute names are specified, these names and their values are displayed in the order specified, provided that each of the specified attribute names is valid. When no attribute names are specified:

- the **-A p** | **d** | **b** flag controls whether persistent attributes or dynamic attributes or both — and their values — are displayed.
- only attributes that are defined as **public** are displayed. Use the **-p** flag to override this default.

For best performance, specify either the **-A p** flag or only persistent attributes as parameters.

Specify the **-r** flag to display only the resource handles associated with the resources for the specified resource class.

To display a list of the attributes and values for a resource class, specify the **-c** flag.

By default, the resource attributes and values are displayed in long format. Use the **-t**, **-d**, or **-D** flag to display the resources in table format or delimiter-formatted output.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

The **lsrsrc** command does not list any attributes that have a datatype defined as **ct_none** (**Quantum**, for example). RMC does not return attribute values for attributes that are defined as **Quantum**. To list attribute definitions, use the **lsrsrcdef** command.

Flags

-a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lsrsrc -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A p | d | b
Specifies an attribute type. By default only persistent attributes are displayed. This flag can be used only when no attribute names are specified on the command line.

p Displays only persistent attributes.

d Displays only dynamic attributes.

b Displays both persistent and dynamic attributes.

For best performance, specify the **-A p** flag.

-c Displays the attributes for the resource class. This flag overrides the **-r** flag.

-C *domain_name...*
Displays the class attributes of a globalized resource class on one or more RSCT peer domains that are defined on the management server. Globalized classes are used in peer domains and management domains for resource classes that contain information about the domain. To display class attributes of a globalized resource class on all peer domains defined on the management server, use the **-c** flag with the **-a** flag instead of **-C**. The command returns the name of the peer domain in the form of an attribute **ActivePeerDomain**. This is not an actual attribute, but is presented as such to indicate which peer domain is being displayed.

-d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-i Generates a template of *resource_data_input_file* that can then, after appropriate editing, be used as input to the **mkrsrc** command. The output is displayed in long (stanza) format. All required and optional attributes that can be used to define a resource are displayed. The attribute data type is displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the **lsrsrc** command be directed to a file. This flag overrides the **-s** and **-A d** flags.

-l Specifies long formatted output. Each attribute is displayed on a separate line. This is the default display format. If the **lsrsrc** command is issued with the **-l** flag, but without a resource class name, the **-l** flag is ignored when the command returns the list of defined resource class names.

-N { *node_file* | "-" }
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*.
- A number sign (#) in column 1 indicates that the line is a comment.
- Any blank characters to the left of a node name are ignored.
- Any characters to the right of a node name are ignored.

Use **-N** "-" to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-p *property*
Displays attributes with the specified *property*. By default, only public attributes are displayed. To

display all of the attributes regardless of the property, use the **-p 0** flag. Use this flag in conjunction with the **-A** flag when no attributes are specified on the command line.

Persistent attribute properties:

0x0001 read_only

0x0002 reqd_for_define (required)

0x0004 inval_for_define (not valid)

0x0008 option_for_define (optional)

0x0010 selectable

0x0020 public

Dynamic attribute properties:

0x0020 public

A decimal or hexadecimal value can be specified for the property. To display attributes and their values for all attributes that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the **-p** flag. For example, to display attributes and their values for all persistent attributes that are either **reqd_for_define** or **option_for_define**, enter:

```
lsrsrc -p 0x0a
```

-r Displays the resource handles for the resources that match the specified selection string or all resources when no selection string is specified.

-s *"selection_string"*
Specifies a selection string. All selection strings must be enclosed within either double or single quotation marks. If the selection string contains double quotation marks, enclose the entire selection string in single quotation marks. For example:

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

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

-t Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x Suppresses header printing.

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

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

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

Parameters

resource_class
Specifies the name of the resource class with the resources that you want to display.

attr... Specifies one or more attribute names. Both persistent and dynamic attribute names can be specified to control which attributes are displayed and their order. Zero or more attributes can be specified. Attributes must be separated by spaces.

Security

The user needs read permission for the *resource_class* specified in **lsrsrc** to run **lsrsrc**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide*

for information about the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all of the resource classes, enter:

   ```
   lsrsrc
   ```

   The output will look like this:

   ```
   class_name
   "IBM.Association"
   "IBM.Condition"
   'IBM.EventResponse"
   "IBM.Host"
   "IBM.Ethernet"
   "IBM.TokenRing"
   ...
   ```

2. To list the persistent attributes for resource IBM.Host that have 4 processors, enter:

   ```
   lsrsrc -s "NumProcessors == 4" -A p -p 0 IBM.Host
   ```

   The output will look like this:

   ```
   Resource Persistent Attributes for: IBM.Host
   resource 1:
           Name           = "c175n05.ppd.pok.ibm.com"
           ResourceHandle = "0x4008 0x0001 0x00000000 0x0069684c 0x0d7f55d5 0x0c32fde3"
           Variety        = 1
           NodeList       = {1}
           NumProcessors  = 4
           RealMemSize    = 1073696768
   ```

3. To list the public dynamic attributes for resource IBM.Host on node 1, enter:

   ```
   lsrsrc -s 'Name == "c175n05.ppd.pok.ibm.com"' -A d IBM.Host
   ```

   The output will look like this:

   ```
   Resource Dynamic Attributes for: IBM.Host
   resource 1:
           ProcRunQueue        = 1.03347987093142
           ProcSwapQueue       = 1.00548852941929
           TotalPgSpSize       = 65536
           TotalPgSpFree       = 65131
           PctTotalPgSpUsed    = 0.61798095703125
           PctTotalPgSpFree    = 99.3820190429688
           PctTotalTimeIdle    = 0
           PctTotalTimeWait    = 51.5244382399734
           PctTotalTimeUser    = 12.8246006482343
           PctTotalTimeKernel  = 35.6509611117922
           PctRealMemFree      = 66
           PctRealMemPinned    = 4
           RealMemFramesFree   = 173361
           VMPgInRate          = 0
           VMPgOutRate         = 0
           VMPgFaultRate       = 0
            ...
   ```

4. To list the Name, Variety, and ProcessorType attributes for the IBM.Processor resource on all the online nodes, enter:

   ```
   lsrsrc IBM.Processor Name Variety ProcessorType
   ```

   The output will look like this:

   ```
   Resource Persistent Attributes for: IBM.Processor
   resource 1:
           Name          = "proc3"
           Variety       = 1
           ProcessorType = "PowerPC_604"
   resource 2:
           Name          = "proc2"
   ```

```
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 3:
        Name          = "proc1"
        Variety       = 1
        ProcessorType = "PowerPC_604"
resource 4:
        Name          = "proc0"
        Variety       = 1
        ProcessorType = "PowerPC_604"
```

5. To list both the persistent and dynamic attributes for the resource class IBM.Condition, enter:

   ```
   lsrsrc -c -A b -p 0 IBM.Condition
   ```

   The output will look like this:

   ```
   Resource Class Persistent and Dynamic Attributes for: IBM.Condition
   resource 1:
           ResourceType = 0
           Variety      = 0
   ```

6. To list the nodes in the cluster that have at least four processors, using the **/tmp/common/node_file** file:

   ```
   # common node file
   #
   node1.ibm.com      main node
   node2.ibm.com      main node
   node4.ibm.com      backup node
   node6.ibm.com      backup node
   #
   ```

   as input, enter:

   ```
   lsrsrc -s "NumProcessors >= 4" -N /tmp/common/node_file -t IBM.Host \
   Name NumProcessors
   ```

   The output will look like this:

   ```
   Resource Persistent Attributes for IBM.Host
   Name              NumProcessors
   "node1.ibm.com"   4
   "node2.ibm.com"   4
   ```

Location

/usr/sbin/rsct/bin/lsrsrc

lsrsrcdef Command

Purpose

Displays definition information for a resource or a resource class.

Syntax

For a *resource*...

To display the definition:

lsrsrcdef [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the persistent attribute definitions:

lsrsrcdef -A p [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr*...]

To display the dynamic attribute definitions:

lsrsrcdef -A d [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr...*]

For a *resource class...*

To display the definition:

lsrsrcdef -c [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr...*]

To display the persistent attribute definitions:

lsrsrcdef -c -A p [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr...*]

To display the dynamic attribute definitions:

lsrsrcdef -c -A d [**-p** *property*] [**-e**] [**-s**] [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] *resource_class* [*attr...*]

To display a list of all of the resource class names:

lsrsrcdef

Description

The **lsrsrcdef** command displays the definition of a resource or a resource class or the persistent or dynamic attribute definitions of a resource or a resource class. By default:

- if no *attr* parameters are specified on the command line, this command displays the definitions for **public** attributes. To override this default, use the **-p** flag or specify the name of the attribute you want to display.
- this command does not display attribute descriptions. To display attribute definitions and descriptions, specify the **-e** flag.

Flags

-A p | d
: Specifies the attribute type. You can display either persistent or dynamic attribute definitions. Use this flag with the **-c** flag to display the persistent or dynamic attribute definitions of a resource class.

  p Displays only persistent attributes

  d Displays only dynamic attributes

-c Displays the definition of a resource class definition. To display the persistent attribute definitions for a resource class, specify this flag with the **-A p** flag. To display the dynamic attribute definitions for a resource class, specify this flag with the **-A d** flag.

-d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag to change the default delimiter.

-D *delimiter*
: Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e Specifies expanded format. By default, the descriptions of the definitons are not displayed. Specify this flag to display the definitions and the descriptions.

-i Generates a template of *resource_data_input_file* that can then, after appropriate editing, be used as

input to the **mkrsrc** command. The output is displayed in long (stanza) format. All required and optional attributes that can be used to define a resource are displayed. The attribute data type is displayed as the value in the *attr=value* pairs. It is suggested that when you use this flag, the output of the **lsrsrcdef** command be directed to a file. This flag overrides the **-s** and **-A d** flags.

-l Specifies "long" format — one entry per line. This is the default display format. If the **lsrsrcdef -l** command is issued without a resource class name, this flag is ignored when the command returns the list of defined resource class names.

-p *property*
Displays attribute definitions for attributes with the specified *property*. By default, only the definitions for **public** attributes are displayed. To display all attribute definitions regardless of the property, use the **-p 0** flag.

Persistent attribute properties:

0x0001 read_only

0x0002 reqd_for_define (required)

0x0004 inval_for_define (not valid)

0x0008 option_for_define (optional)

0x0010 selectable

0x0020 public

Dynamic attribute properties:

0x0020 public

A decimal or hexadecimal value can be specified for the property. To request the attribute definitions for all attributes that have one or more properties, "OR" the properties of interest together and then specify the "OR"ed value with the **-p** flag. For example, to request the attribute definitions for all persistent attributes that are either **reqd_for_define** or **option_for_define**, enter:

```
lsrsrcdef -p 0x0a
```

-s Displays the structured data definition. Specify this flag for the structured data definition to be expanded so that each element definition of the structured data attributes is displayed.

-t Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x Suppresses header printing.

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

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

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

Parameters

resource_class
Specifies the name of the resource class with the attribute definitions you want to display.

attr If a *resource_class* parameter is specified, zero or more attribute names can be specified. If no *attr* parameter is specified, the definition for all of the attributes for the resource are displayed. Specify individual attribute names to control which attributes are displayed and their order. Specify only persistent attribute names when the **-A p** flag is used. Specify only dynamic attribute names when the **-A d** flag is used. Attributes must be separated by spaces.

Security

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

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To display the names of all of the resource classes defined on the system, enter:

   ```
   lsrsrcdef
   ```

   The output will look like this:

   ```
   class_name
   "IBM.ATMDevice"
   "IBM.Association"
   "IBM.AuditLog"
   "IBM.AuditLogTemplate"
   "IBM.Condition"
   "IBM.EthernetDevice"
   "IBM.EventResponse"
   ...
   ```

2. To display the resource class definitions for resource IBM.Host, enter:

   ```
   lsrsrcdef -c IBM.Host
   ```

   The output will look like this:

   ```
   Resource Class Definition for: IBM.Host
   resource class 1:
           class_name         = "IBM.Host"
           class_id           = 8
           properties         = {"has_rsrc_insts","mtype_subdivided"}
           display_name       = ""
           description        = ""
           locator            = "NodeList"
           class_pattr_count  = 1
           class_dattr_count  = 3
           class_action_count = 0
           pattr_count        = 6
           dattr_count        = 47
           action_count       = 0
           error_count        = 0
           rsrc_mgr_count     = 1
   rsrc_mgrs 1:
           mgr_name  = "IBM.HostRM"
           first_key = 1
           last_key  = 1
   ```

3. To display the resource class persistent attribute definitions for resource IBM.Host, enter:

   ```
   lsrsrcdef -c -A p -p 0 IBM.Host
   ```

   The output will look like this:

   ```
   Resource Class Persistent Attribute Definitions for: IBM.Host
   attribute 1:
           program_name              = "Variety"
           display_name              = ""
           group_name                = ""
           properties                = {"read_only","inval_for_define"}
           description               = ""
           attribute_id              = 0
           group_id                  = 255
           data_type                 = "uint32"
           variety_list              = {{1..1}}
           variety_count             = 1
           default_value             = 0
   ```

4. To display the resource persistent attribute definitions and descriptions for resource IBM.Host, enter:

```
lsrsrcdef -A p -p 0 -e IBM.Host
```

The output will look like this:

```
Resource Persistent Attribute Definitions for: IBM.Host
attribute 1:
        program_name       = "Name"
        display_name       = "Name"
        group_name         = "General"
        properties         = {"reqd_for_define","public","selectable"}
        description        = "Identifies the current name of the host
                              as returned by command."
        attribute_id       = 0
        group_id           = 0
        data_type          = "char_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = ""
attribute 2:
        program_name       = "ResourceHandle"
        display_name       = "Resource Handle"
        group_name         = "Internal"
        properties         = {"read_only","inval_for_define","selectable"}
        description        = "A globally unique handle that identifies the host.
                              Every resource is assigned a resource handle,
                              which is used internally for identifying and
                              locating each resource.  The resource handle
                              is fixed in size and avoids the problems of
                              name space collisions across different types
                              of resources."
        attribute_id       = 1
        group_id           = 255
        data_type          = "rsrc_handle_ptr"
        variety_list       = {{1..1}}
        variety_count      = 1
        default_value      = "0x0000 0x0000 0x00000000 0x00000000 0x00000000 0x00000000"
attribute 3:
        program_name       = "Variety"
        display_name       = "Variety"
        group_name         = "Internal"
 ...
```

5. To display the public dynamic attributes for resource IBM.Host, enter:

```
lsrsrcdef -A d IBM.Host
```

The output will look like this:

```
Resource Dynamic Attribute Definitions for: IBM.Host
attribute 1:
        program_name           = "ProcRunQueue"
        display_name           = ""
        group_name             = ""
        properties             = {"public"}
        description            = ""
        attribute_id           = 1
        group_id               = 1
        data_type              = "float64"
        variable_type          = 0
        variety_list           = {{1..1}}
        variety_count          = 1
        init_value             = 0
        min_value              = 0
        max_value              = 100
        expression             = "(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)"
        expression_description = ""
        rearm_expression       = "ProcRunQueue < 50"
```

```
        rearm_description      = ""
        PTX_name               = ""
attribute 2:
...
```

Location

/usr/sbin/rsct/bin/lsrsrcdef

mkrsrc Command
Purpose

Defines a new resource.

Syntax

To define a new resource, using data that is...

- entered on the command line:
  mkrsrc [**-a** | **-N** { *node_file* | **"-"** }] **[-v] [-h] [-TV]** *resource_class attr=value...*
- predefined in an input file:
  mkrsrc -f *resource_data_input_file* **[-v]** [**-a** | **-N** { *node_file* | **"-"** }] **[-h] [-TV]** *resource_class*

To display the names and datatypes of the command arguments:

mkrsrc -l [**-h**] *resource_class*

To see examples of the **mkrsrc** command for a resource class:

mkrsrc -e [**-h**] [**-TV**] *resource_class*

Description

The **mkrsrc** command requests that the RMC subsystem define a new resource instance for the class specified by the *resource_class* parameter. At least one persistent attribute name and its value must be specified either as a parameter or by a resource definition file using the **-f** flag.

Before you run **mkrsrc**, you should run the **lsrsrcdef** command to determine which attributes are designated as **reqd_for_define** (required) or **option_for_define** (optional). Only attributes that are designated as **reqd_for_define** or **option_for_define** can be defined using the **mkrsrc** command. The **lsrsrcdef** command also identifies the datatype for each attribute. The value specified for each attribute must match this datatype.

To verify that all of the attribute names that are specified on the command line or in *resource_data_input_file* are defined as persistent attributes and are designated as **reqd_for_define** or **option_for_define**, use the **-v** flag. When the **mkrsrc** command is run with the **-v** flag, the resource is not defined. Instead, the resource attributes are merely verified to be persistent and designated as **reqd_for_define** or **option_for_define**. Once you have run **mkrsrc -v** to verify that all of the attributes that are specified on the command line or in *resource_data_input_file* are valid, you can issue the **mkrsrc** command without the **-v** flag to define the new resource.

If you are running in an RSCT peer domain or on the management server in an RSCT management domain and the resource class management type is subdivided, you can create the same resource on multiple nodes in one of two ways. The first way is to use the **-N** *node_file* flag to indicate that the node names to create the resources on are in a file. Use **-N "-"** to read the node names from standard input. The second way is to specify multiple node names in the **NodeNameList** resource attribute. The **NodeNameList** attribute defines where the resource is created when a cluster is present. If the

NodeNameList attribute is not used, the resource is created on the local node. To find out if a resource class management type is subdivided, enter **lsrsrcdef -c** *resource_class* | **grep properties**.

Flags

-e Displays examples of **mkrsrc** command-line input for:

1. required attributes only
2. required and optional attributes

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

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

-N { *node_file* **| "-" }**
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "-"** to read the node names from standard input.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If the resource class management type of the resource that is to be defined is subdivided and **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-v Verifies that all of the attribute names specified on the command line or in the input file are defined as persistent attributes and are designated as **reqd_for_define** or **option_for_define**. The **mkrsrc** command does *not* define any resources when you use this flag.

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

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

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

Parameters

resource_class
Specifies the resource class name of the resource to be defined.

attr=value...
Specifies the attributes of the resource being defined. When defining a new resource instance, there are specific required attributes for each resource that must be defined. These attributes can be specified as parameters on the command line or defined in an input file by using the **-f** flag.

attr The name of a persistent attribute for this resource. This attribute must be designated as **reqd_for_define** or **option_for_define**. Use the **lsrsrcdef** command to check the designation.

value The value for this persistent attribute. The data type for this value must match the defined data type for the value of this attribute. Use the **lsrsrcdef** command to verify the data type for each attribute.

Security

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

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the **rsct.rmc** fileset for the AIX® operating system.

Standard Output

- All command output is written to standard output.
- When the **-h** flag is specified, this command's usage statement is written to standard output.
- When the **-V** flag is specified, this command's verbose messages (if there are any available) are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To create a new resource in the **IBM.Host** class, assuming you already know which persistent attributes are required when defining a resource of this class, enter:

   ```
   mkrsrc IBM.Host Name=c175n05
   ```

2. To create a new resource in the **IBM.Processor** class by first generating a template to aid in the defining of these resources, enter:

   ```
   lsrsrcdef -i IBM.Processor > /tmp/IBM.Processor.rdef
   ```

   Then, edit the file **/tmp/IBM.Processor.rdef** and enter values for all of the attributes, substituting the type for an appropriate value, or leaving it blank for the default value.

   Finally, enter:

   ```
   mkrsrc -f /tmp/IBM.Processor.rdef IBM.Processor
   ```

3. To create two new **IBM.Host** resources using the information defined in file **/tmp/IBM.Host.rdef**, enter:

   ```
   mkrsrc -f /tmp/IBM.Host.rdef IBM.Host
   ```

   where the file **/tmp/IBM.Host.rdef** looks like this:

   ```
   PersistentResourceAttributes::
   resource 1:
       Name        = c175n04

   resource 2:
       Name        = c175n05
   ```

4. This example creates a new resource in the **IBM.Foo** class. In this class, **Name** and **NodeList** are required attributes. The **Binary**, **SD**, **StringArray**, and **SDArray** attributes are optional. This example shows how to enter the more difficult data types from the command line. The data types for the optional attributes (**Binary**, **SD**, **StringArray**, and **SDArray**) are self-explanatory. Enter:

   ```
   mkrsrc IBM.Foo Name=c175n05 \
   NodeList={1} \
   Binary="0xaabbccddeeff00" \
   SD='[testing123,1,{2,4,6}]' \
   StringArray='{"testing 1 2 3",testing123,"testing 1 2 3"}' \
   SDArray='{["testing 1 2 3",1,{1,3,5}],[testing,2,{2,4,6}]}'
   ```

5. To create resources for the IBM.Example class on multiple nodes in a peer domain, run this command:

   ```
   mkrsrc -N /u/joe/common_node_file  IBM.Example  Name=Example_bar1 \
   Binary="0xaabbccddeeff00"
   ```

   where the contents of **/u/joe/common_node_file** look like this:

   ```
   # common node file
   #
   node1.ibm.com      main node
   node2.ibm.com      main node
   node4.ibm.com      backup node
   node6.ibm.com      backup node
   #
   ```

6. To create resources of the IBM.Example class on multiple managed nodes in a management domain, run this command on the management server:

```
mkrsrc IBM.Example  Name=Example_bar1 Binary="0xaabbccddeeff00" \
NodeNameList='{"mgnode1.ibm.com","mgnode2.ibm.com"}'
```

where the contents of **/u/joe/common_node_file** look like this:

```
# common node file
#
node1.ibm.com      main node
node2.ibm.com      main node
node4.ibm.com      backup node
node6.ibm.com      backup node
#
```

Note: As discussed in the **rmccli** general information file, attribute values for certain data types (structured data, array of structured data, and arrays containing strings enclosed in double quotation marks) should be enclosed in single quotation marks.

Location

/usr/sbin/rsct/bin/mkrsrc

refrsrc Command
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.

Flags

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

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

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

Parameters

resource_class
Specifies the resource class name.

Security

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

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

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

Standard Error

All trace messages are written to standard error.

Examples

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

```
refrsrc IBM.FileSystem
```

Location

/usr/sbin/rsct/bin/refrsrc

resetrsrc Command
Purpose

Resets a resource that is, forces the resource to move to the offline state.

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 data types 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 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 dynamic attribute **OpState** to determine when the resource is forced offline. Register an event for the resource, specifying the **OpState** attribute, to know when the resource is 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 data type must match the definition of the argument data type.

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

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

Flags

-f *resource_data_input_file*
Specifies the name of the file that contains resource argument information. The following contents of the file is displayed:

```
PersistentResourceArguments::

argument1 = value1

argument2 = value2
```

-l Lists the command arguments and data types. Some resource managers accept additional arguments that are passed to the offline request. Use this flag to list any defined command arguments and the data types 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 usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages (if there are any available) 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 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 has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon 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 usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 No resources were found that match the specified selection string.

Security

You need 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 *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the **rsct.core.rmc** fileset for AIX and **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/bin/resetrsrc

Examples

Suppose that 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
```

rmrsrc Command

Purpose

Removes a defined resource.

Syntax

To remove one or more resource.

- 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.*

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 using the format: **"0x***nnnn* **0x***nnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn* **0x***nnnnnnnn***"**, where *n* is any valid hexadecimal digit. 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.

Parameters

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

Security

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

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 No resources were found that match the selection string.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

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

Standard Error

All trace messages are written to standard error.

Examples

1. To remove the resource with the Name c175n05 from resource class IBM.Host, enter:

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

2. To remove the resource 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
   ```

Location

/usr/sbin/rsct/bin/rmrsrc

runact Command

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 data type for each input element. The value specified for each input element must match this data type.

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:

```
"0xnnnn 0xnnnn 0xnnnnnnnnn 0xnnnnnnnnn 0xnnnnnnnnn 0xnnnnnnnnn"
```

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 *Administering RSCT*.

-t Specifies table format. Each attribute is displayed in a separate column, with one resource per line.

-x Suppresses header printing.

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

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

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

Parameters

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.

Security

This command requires **root** authority.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the Resource Monitoring and Control (RMC) daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

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

Standard Error

All trace messages are written to standard error.

Examples

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

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

The output will look like this:

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

Location

/usr/sbin/rsct/bin/runact
Contains the **runact** command

startrsrc Command
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 data types 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 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 dynamic attribute **OpState** of the resource to determine when the resource is 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 data type must match the definition of the argument data type.

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

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

Flags

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

```
PersistentResourceArguments::

argument1 = value1

argument2 = value2
```

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

-n *node_name*
Specifies the name of the node where the resource is to be brought online. *node_name* is a **NodeNameList** attribute value. Use this flag to bring a floating resource online on a different node if the node where it was online might be down.

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 usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages (if there are any available) 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 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 usage statement is written to standard output. When the **-V** flag is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 No resources were found that match the specified selection string.

Security

You need 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 *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the **rsct.core.rmc** fileset for AIX operating system and **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows operating systems, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/bin/startrsrc

Examples

Suppose that 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"
```

stoprsrc Command

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 data types 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 performed by the appropriate resource manager.

To stop one or more resources, use the **-s** flagoption 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* flagoption 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** flagoption to specify the resource handle that represents that specific resource.

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

If Cluster Systems Management (CSM) is installed on your system, yYou 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 dynamic attribute **OpState** to determine when the resource is taken offline. Register an event for the resource, specifying the **OpState** attribute, to know when the resource is 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 datatype must match the definition of the argument datatype.

Command arguments are optional. If any *arg*=*value* pairs are entered, there should be one *arg*=*value* pair for each command argument 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 argument information. The contents of the file would look like this:

```
PersistentResourceArguments::

argument1 = value1

argument2 = value2
```

-l Lists the command arguments and data types. Some resource managers accept additional arguments that are passed to the offline request. Use this flagoption to list any defined command arguments and the data types 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 usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages (if there are any available) 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 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** flagoption is specified, this command usage statement is written to standard output. When the **-V** flagoption is specified, this command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flagoption 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

You need 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 *Administering RSCT* guide for information about the ACL file and how to modify it.

Implementation specifics

This command is part of the **rsct.core.rmc** fileset for the AIX operating system. **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows operating systems, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/bin/stoprsrc

Examples

Suppose that 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"
   ```

RMC information files

resource_data_input Information File

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 about the command line would be too cumbersome or 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. The resource data input file, which must be in POSIX format, 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 starting 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 starting a 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 is 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 **chrsrc** command 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 can 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.
  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 might not be the same format used on the command line. The shell you choose to run the commands in has its own rules regarding 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 information is part of the `rsct.core.rmc` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is i386, ppc, ppc64, s390, or x86_64.

Location

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

Examples

1. This sample **mkrsrc** command:

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

   uses the sample input file `/tmp/my_resource_data_input_file` for the `IBM.Example` 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.Example 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/Example/ch_resources -s 'Name == "c175n05"' IBM.Example
```

uses the sample input file `/tmp/Example/ch_resources` to change the attribute values of existing `IBM.Example` 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.Examplebar
   ```

   shows the optional command arguments:

   ```
   rmrsrc IBM.Examplebar ExampleInt32=int32 ExampleUint32=uint32
   ```

4. This sample **rmrsrc** command:

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

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

   ```
   PersistentResourceArguments::

   # Specifying command arguments when running rmrsrc command.

   resource 1:

    ExampleInt32      =   1

           ExampleUint32     =   0
   ```

rmccli information file

Purpose

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

Description

The general information about RMC and related commands, including data types, terminology, and references to related information follows.

Command structure and use

The RMC commands might be grouped into categories that represent the different operations that can be run 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 flag is the default display format.

   For example, the command:

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

   produces the following output:

```
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 the following output:
   ```
   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 the following output:
   ```
   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 the following output:
   ```
   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 might 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*nnnnnnnnnnnnnnnnnnnn*..."
- 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 about how to specify selection strings, see the *Administering RSCT* 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 structured data (SD) is supplied 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 arrays of structured data or arrays that contain strings enclosed in quotation marks are supplied 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`.

Data output formatting

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

- A description attribute that equals the string "This is a string that contains white space" is displayed in the 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 that contains 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 by 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, an 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 stand-alone daemon, or it can be integrated into an application or a subsystem directly.

To see all of the resource classes that are defined in the system, run the **lsrsrc** command without any flags or parameters. To see all of the resources that are 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.

Flags

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages (if there are any available) 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 on tracing. Trace messages are not translated. Use the **-V** flag, which indicates "verbose" mode, to see more information about the command. Verbose messages (if there are any available) 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 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. The CT_IP_AUTHENT environment variable is valid, if the CT_CONTACT environment variable 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 usage statement is written to standard output. When the `-V` flag is specified, these command verbose messages (if there are any available) are written to standard output.

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 No resources were found that match the specified selection string.

Security

Permissions are specified in the access control list (ACL) file on the contacted system.

Implementation specifics

This information is part of the `rsct.core.rmc` fileset for AIX and `rsct.core-3.1.0.0-0.`*platform*`.rpm` package for Linux, Solaris, and Windows, where *platform* is `i386`, `ppc`, `ppc64`, `s390`, or `x86_64`.

Location

/usr/sbin/rsct/man/rmccli
`/usr/sbin/rsct/man/rmccli.7` - For Linux platform.

snmptrapd.conf Information File
Purpose

Contains entries to call the **trap2rmc** command when the traps are received.

Description

The `snmptrapd.conf` configuration file contains entries to call the **trap2rmc** command when traps are received.

Implementation specifics

This command is part of the **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/share/snmp/snmptrapd.conf
Contains the `snmptrapd.conf` configuration file.

RSCT peer domain configuration commands

Using configuration resource manager commands, you can list and modify various aspects of the peer domain configuration. For example, you can create a peer domain, add or remove nodes to an existing peer domain, take a node or entire peer domain offline, and so on.

addrpnode Command

Purpose

Adds one or more nodes to a peer domain definition.

Syntax

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

addrpnode [**-c**] { **-f** | **-F** { *file_name* | **"-"** } } [**-h**] [**-TV**]

| **addrpnode** [**-c**] [**-h**] [**-TV**] *node_name1* [*@host_name1*] [*node_name2* [*@host_name2*] ...]

Description

Before running the addrpnode command:

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 the **addrpnode** command 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 be online only in one peer domain. To add one or more nodes to the peer domain, more than half of the nodes must be online.

To enable the **addrpnode** command 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 so, use the **startrpnode** command.

Flags

-c Continues processing the command while 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 runs on the other nodes, even if it fails on one node.

-f | -F { *file_name* **| "-" }**
Specifies that node names are read from a file or from standard input.

Use **-f** *file_name* or **-F** *file_name* to read the node names from a file. Use **-f "-"** or **-F "-"** to specify **STDIN** as the input file.

Notes:

- 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 that are listed in *file_name*:

- are Group Services group leader candidates.
- are used for quorum decisions.
- have access to the peer domain tiebreaker mechanism.

You can customize node characteristics by using an at sign (@) control character followed by one or more of these special characters:

P | p Specifies that the node is a Group Services group leader candidate.

Q | q Specifies that the node is a quorum node.

B | b Specifies that the node has access to the peer domain tiebreaker mechanism. **B** or **b** can be specified only for quorum nodes.

! Specifies that the node does not have a certain characteristic. For example, **!Q** indicates that the node is not a quorum node.

When customizing node characteristics, consider the following points (where **x** is **P**, **Q**, or **B**):

- Use only one @ control character per line, followed immediately by one or more special characters, after the node name and before any comments.
- Do not specify **!QB** for a node; it results an error.
- If you use a node number, add it after the node name and before any comments. The node number can precede or follow the node characteristic specifications.
- If **x** is specified for one or more nodes and **!x** is not specified for any nodes, the nodes that do not have an **x** specified are assumed to have a value of **!x**.
- If **!x** is specified for one or more nodes and **x** is not specified for any nodes, the nodes that do not have an **!x** specified are assumed to have a value of **x**.
- If **x** and **!x** are specified for different nodes in the same node file, all of the nodes in the file must have a specification of **x** or **!x**.

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

Parameters

node_name1 **[***node_name2* **...]**
Specifies the node (or nodes) 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.

| *node_name1***[@***host_name1***] [***node_name2***[@***host_name2***] ...]**

Specifies the nodes that need to be added to RPD by using the node name along with the host name for each node. The *node_name1* parameter corresponds to a label but the *host_name1* parameter is either the IP address or a long or short version of the DNS host name. The host name must be a valid value that can be contacted or pinged.

If the *HostName* parameter is not specified and only *Name* parameter is specified for the **addrpnode** command, the *HostName* parameter is set as the *Name* parameter. In this case, the *Name* parameter must resolve to IP address or long or short version of the DNS host name.

To add a node to the existing peer domain, use the following command:

```
addrpnode node_name3@host_name3
```

You can also run the **addrpnode -f /home/nodelst** command, where */home/nodelst* has node names as *node_name3@host_name3.in.ibm.com*.

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. It 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. It gives the online nodes the necessary authority to perform operations on the nodes to be added.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

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.

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, the command usage statement is written to standard output. All verbose messages are written to standard output.

Standard Error

All trace messages are written to standard error.

Examples

1. To add the nodes **node_name2** and **node_name3** to the peer domain **ApplDomain**, where **node_name1** is already defined and online on the peer domain**ApplDomain**, run command on **node_name1**:

   ```
   addrpnode node_name2 node_name3
   ```

2. To add the nodes **node_name2** and **node_name3** along with the host names to the peer domain **ApplDomain**, where **node_name1** is already defined and online on the peer domain **ApplDomain**, run command on **node_name1**:

   ```
   addrpnode node_name2@host_name2 nodeC_name3@host_name3
   ```

Location

```
/usr/sbin/rsct/bin/addrpnode
```

chcomg Command

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*] [**-g** *grace*] [**-t** *priority*] [**-b**] [**-r**] [**-x b** | **r** | **br**] [**-e** *NIM_path*] [**-m** *NIM_parameters*] [**-N** *UseForNodeMembership*] [**-h**] [**-TV**] *communication_group*

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

chcomg [**-i h:***heartbeat_interface1*[**:***node1*] [**,***heartbeat_interface2*[**:***node2*]...] | **-S h:"***heartbeat_interface_selection_string***"**] [**-h**] [**-TV**] *communication_group*

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

chcomg [**-i n:***network_interface1*[**:***node1*][**,***network_interface2*[**:***node2*]...] | **-S n:"***network_interface_selection_string***"**] [**-6**] [**-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 nodes must be online to change a communication group in the domain.

Flags

-s *sensitivity*
Specifies the heartbeat sensitivity. This is the number of missed heartbeats that constitute a failure. The sensitivity is an integer 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**.

-g *grace*
Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

0 Specifies that the grace period is disabled.

-1 | d Specifies that the topology services subsystem controls the grace period. This is the default value.

-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 h | 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. Specify **-i h** for heartbeat interface resources or **-i n** for network interface resources. By default, the **-i n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-i n** flag adds network interface resources that have IPv6 addresses to *communication_group*.

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

-S h | n: "*network_interface_selection_string***"**
Assigns this communication group to the interface specified by the network interface selection string. Specify **-S h** for heartbeat interfaces or **-S n** for network interfaces. By default, the **-S n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-S n** flag adds network interface resources that have IPv6 addresses to *communication_group*.

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.

-N *UseForNodeMembership*
Specifies whether group services use the communication group in calculating node membership. Sets the **UseForNodeMembership** persistent resource attribute for the communication group resource. Valid values are:

0 Indicates that, regardless of the results of liveness checks run on **NetworkInterface** resources that are members of this communication group, group services do not use those results in calculating whether the node owning the interfaces is online.

1 Indicates that group services use the results of liveness checks run on the **NetworkInterface** resources in calculating the online state of their owning nodes.

-6 Specifies that IPv6 addresses represented as resources on each interface have their communication group changed to the one specified. IPv4 addresses represented as resources on the interfaces are unaffected.

By default (without the **-6** flag specified), the inverse is true. Only IPv4 addresses represented as resources on the interface have their communication group changed.

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

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

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

Parameters

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

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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be changed.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

In these examples, 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 heartbeat interface resource named **hbi0** on **nodeC**, run this command on **nodeA**:

   ```
   chcomg -i h:hbi0:nodeC ComGrp1
   ```

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

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

7. To assign the communication group **ComGrp1** for **ApplDomain** to the heartbeat interface resource that uses the subnet 9.345.67.812, run this command on **nodeA**:

   ```
   chcomg -S h:"Subnet == '9.345.67.812'" ComGrp1
   ```

8. 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
   ```

9. To change the communication group **ComGrp1** for **ApplDomain** to a period of 500 milliseconds, run this command on **nodeA**:

   ```
   chcomg -p 0.5 ComGrp1
   ```

Location

/usr/sbin/rsct/bin/chcomg

forcerpoffline Command

Purpose

Forces a peer domain to be offline.

Syntax

forcerpoffline [**-h**] *domain_name*

Description

Attention: Use this command with extreme caution.

The **forcerpoffline** command must be used only if a node is in a pending online state and you are unable to bring it online using the **startrpdomain** command. This scenario can 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 the **forcerpoffline** command. As a result of running the **forcerpoffline** command, the configuration resource manager subsystem (**IBM.ConfigRM**) and the RMC subsystem (**ctrmc**) are recycled.

Parameters

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

Flags

-h Writes the command 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 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 **rsct.basic.rte** fileset for AIX and **rsct.basic-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/bin/forcerpoffline

lscomg Command

Purpose

Displays information about the communication groups of a peer domain.

Syntax

lscomg [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-i**] [**-h**] [**-TV**] [*communication_group*]

Description

The **lscomg** command displays information about the communication groups that are defined to the online peer domain on which the command runs. If you specify the name of a communication group, the **lscomg** command displays information about that communication group only.

Some of the communication group information that is displayed follows:

Field	Description
Name	The name of the communication group
Sensitivity	The number of missed heartbeats that constitute a failure
Period	The number of seconds between heartbeats
Priority	The relative priority of the communication group
Broadcast	Indicates whether broadcast should be used if it is supported by the underlying media
SourceRouting	Indicates whether source routing should be used if it is supported by the underlying media
NIMPath	The path to the Network Interface Module (NIM) that supports the adapter types in the communication group
NIMParameters	The NIM start parameters

Interface resources

Use the **-i** flag to display information about the interface resources that refer to *communication_group*.

For IP communication groups (**MediaType** = **1**), **lscomg -i** displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.
NodeName	The host name of the interface resource that refers to *communication_group*.
IPAddress	The IP address of the interface resource that refers to *communication_group*.
SubnetMask	The subnet mask of the interface resource that refers to *communication_group*.
Subnet	The subnet of the interface resource that refers to *communication_group*

For disk heartbeating (**MediaType** = **2**) and other non-IP types of communication groups (**MediaType** = **0**), **lscomg -i** displays the following information:

Field	Description
Name	The name of the interface resource that refers to *communication_group*.
NodeName	The host name of the interface resource that refers to *communication_group*.
DeviceInfo	Information about the device.
MediaType	The type of interfaces that make up this communication group.

Flags

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default format.

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x Excludes the header (suppresses header printing).

-i Displays information about the interface resource that refers to *communication_group*.

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

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

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

Parameters

communication_group
Specifies the name of the communication group about which you want to display information. You can specify a communication group name or a substring of a communication group name for this parameter. If you specify a substring, the command displays information about any defined communication group with a name that contains the substring.

Security

The user of the **lscomg** command needs read permission for the **IBM.CommunicationGroup** resource class. Read permission for the **IBM.NetworkInterface** resource class is required to display the network interface information. By default, **root** on any node in the peer domain has read and write access to these resource classes through the configuration resource manager.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The communication group definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

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

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

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

   ```
   lscomg
   ```

   The following output is displayed:

```
Name  Sensitivity  Period  Priority  Broadcast  SourceRouting  NIMPath                      NIMParameters
ComG1      2          2        1        no          yes        /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 following output is displayed:

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

Location

/usr/sbin/rsct/bin/lscomg

lsrpdomain Command

Purpose

Displays peer domain information for the node.

Syntax

lsrpdomain [**-o** | **-O**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**] [*peer_domain*]

Description

The **lsrpdomain** command displays information about the peer domains that the node where the command runs belongs to. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify the name of a peer domain, the command displays information about that peer domain only. The **-o** and **-O** flags also limit the information this command displays. The **-o** flag displays information only about the online peer domain. The **-O** flag displays information only about peer domains that are offline.

By default, the **lsrpdomain** command displays information in table format (**-t**).

Some of the peer domain information that is displayed follows:

Field	Description
Name	The name of the peer domain.
RSCTActiveVersion	The version of RSCT that is active in the peer domain.
MixedVersions	Indicates whether more than one version of RSCT is active in the peer domain.
TSPort	The topology services port number.
GSPort	The group services port number.
OpState	The current state of the peer domain.

Flags

-o Displays information about the node's online peer domain.

-O Displays information about peer domains that are offline for the node.

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default.

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-x Excludes the header (suppresses header printing).

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

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

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

Parameters

peer_domain
Specifies the name of the peer domain about which you want to display information. You can specify a peer domain name or a substring of a peer domain name for this parameter. If you specify a substring, the command displays information about any defined peer domain with a name that contains the substring.

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on the node for which the peer domain information is requested.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To display general information about the peer domains to which **nodeA** belongs, run this command on **nodeA**:

   ```
   lsrpdomain
   ```

   The output will look like this:

   ```
   Name            OpState         RSCTActiveVersion       MixedVersions   TSPort  GSPort
   ApplDomain      Online          2.5.0.0                  No              12347   12348
   ```

2. To display general information about the peer domains to which **nodeA** belongs, with the default delimiter (but without the heading), run this command on **nodeA**:

   ```
   lsrpdomain -xd
   ```

   The output will look like this:

```
ApplDomain:Online:2.5.0.0:No:12347:12348:
```

3. To display general information about the peer domains to which **nodeA** belongs, in long format, run this command on **nodeA**:

```
lsrpdomain -l
```

The output will look like this:

```
Name              = ApplDomain
OpState           = Online
RSCTActiveVersion = 2.5.0.0
MixedVersions     = No
TSPort            = 12347
GSPort            = 12348
```

Location

/usr/sbin/rsct/bin/lsrpdomain

lsrpnode Command

Purpose

Displays information about one or more of the nodes that are defined in the online peer domain.

Syntax

lsrpnode [[**-i**] [**-l** | **-t** | **-d** | **-D** *delimiter*] **-o** | **-O** | **-L**] **-P**] **-Q**] [**-B**] [**-x**] [**-h**] [**-TV**] [*node_name*]

lsrpnode -p *peer_domain* [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-TV**]

Description

The **lsrpnode** command displays information about one or more of the nodes that are defined in the online peer domain. Use the command's flags and parameters to specify which information you want to display and how you want to display it. When you specify a node name, the command displays information about that node only.

The **-o**, **-O**, and **-L** flags also limit the information this command displays. The **-o** flag displays information about nodes that are online. The **-O** flag displays information about nodes that are offline. The **-L** flag displays information about the local node, which is the node the command runs on.

The **-P** flag displays additional node configuration information related to group services group leader selection. The **-Q** flag displays additional node configuration information related to quorum decisions. The **-B** flag displays additional node configuration information related to the tiebreaker mechanism.

By default, the **lsrpnode** command displays information in table format (**-t**).

Some of the node information that is displayed follows:

Field	Description
Name	The name of the node in the peer domain.
OpState	The operational state of the node.
RSCTVersion	The version of RSCT that is active in the node.

The following fields are displayed when you specify the **-i** flag:

Field	Description
NodeNum	The node number used by topology services and group services. This number is unique within the cluster.
NodeID	The unique node identifier.

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

Field	Description
Preferred	Indicates whether the node is a group services group leader candidate.

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

Field	Description
Quorum	Indicates whether the node participates in quorum decisions.

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

Field	Description
Tiebreaker	Indicates whether the node has access to the peer domain's tiebreaker mechanism.

See the *Administering RSCT* guide for information about group services group leader selection, quorum decisions, and the tiebreaker mechanism.

Flags

-d Displays the information using delimiters. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default colon (**:**) — when the information you want to display contains colons, for example. You can use this flag to specify a delimiter of one or more characters.

-i Displays the node number and node ID for the node. The node number is used by topology services and group services and is unique within the cluster. The node ID is the unique node identifier.

-l Displays the information on separate lines (long format).

-L Displays information about the local node only, which is the node that the command runs on.

-o Displays information about the nodes that are online in the peer domain.

-O Displays information about the nodes that are offline in the peer domain.

-p *peer_domain*
Displays information about nodes defined in an *offline* peer domain that the local node belongs to. (By default, the **lsrpnode** command displays information about the nodes that are defined in the domain where you are currently *online*.) However, this information might not reflect changes that are made to the domain after the local node is taken offline, because an offline node might not have the latest configuration.

The **-p** flag ignores the **CT_CONTACT** environment variable. You must have root access to use the **-p** flag.

-P Indicates whether the node is a group services group leader candidate. **yes** is displayed if the node can be a group services group leader. **no** is displayed if the node cannot be a group services group leader. See the *Administering RSCT* for more information about group services group leader selection.

-Q Indicates whether the node participates in quorum decisions. yes is displayed if the node

participates in quorum decisions. no is displayed if the node does not participate in quorum decisions. See the *Administering RSCT* for more information on quorum decisions.

-B Indicates whether the node has access to the peer domain's tiebreaker mechanism. yes is displayed if the node has access to the peer domain's tiebreaker mechanism. no is displayed if the node does not have access to the peer domain's tiebreaker mechanism. See the *Administering RSCT* for more information on the tiebreaker mechanism.

-t Displays the information in separate columns (table format). This is the default format.

-x Excludes the header (suppresses header printing).

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

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

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

Parameters

node_name
Specifies the name of the node about which you want to display information. You can specify a node name or a substring of a node name for this parameter. If you specify a substring, the command displays information about any defined node with a name that contains the substring.

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

   ```
   lsrpnode
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion
   nodeA   Online    3.1.4.0
   nodeB   Online    3.1.4.0
   nodeC   Offline   3.1.4.0
   ```

2. To display general information about the nodes in the online peer domain that **nodeA** belongs to, with the default delimiter (but without the heading), run this command on **nodeA**:

   ```
   lsrpnode -xd
   ```

   The output will look like this:

   ```
   nodeA:Online:3.1.4.0:
   nodeB:Online:3.1.4.0:
   nodeC:Offline:3.1.4.0:
   ```

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

   ```
   lsrpnode -l
   ```

   The output will look like this:

   ```
   Name        = nodeA
   OpState     = Online
   RSCTVersion = 3.1.4.0

   Name        = nodeB
   OpState     = Online
   RSCTVersion = 3.1.4.0

   Name        = nodeC
   OpState     = Offline
   RSCTVersion = 3.1.4.0
   ```

4. To display general information about the nodes in the online peer domain that **nodeA** belongs to, including the node number and node ID, run this command on **nodeA**:

   ```
   lsrpnode -i
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   NodeNum   NodeID
   nodeA   Online    3.1.4.0       2         40a514bed9d82412
   nodeB   Online    3.1.4.0       1         47fe57098f4ec4d9
   ```

5. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the preferred group services group leader information, run this command on **nodeA**:

   ```
   lsrpnode -P
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Preferred
   nodeA   Online    3.1.4.0       yes
   nodeB   Online    3.1.4.0       no
   ```

6. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including the quorum information, run this command on **nodeA**:

   ```
   lsrpnode -Q
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Quorum
   nodeA   Online    3.1.4.0       no
   nodeB   Online    3.1.4.0       yes
   nodeC   Online    3.1.4.0       yes
   ```

7. To display general information about the nodes in the online peer domain to which **nodeA** belongs, including quorum and tiebreaker information, run this command on **nodeA**:

   ```
   lsrpnode -QB
   ```

   The output will look like this:

   ```
   Name    OpState   RSCTVersion   Quorum  Tiebreaker
   nodeA   Online    3.1.4.0       no    no
   nodeB   Online    3.1.4.0       yes   yes
   nodeC   Online    3.1.4.0       yes   yes
   ```

Location

/usr/sbin/rsct/bin/lsrpnode

mkcomg Command

Purpose

Creates a new communication group definition for a peer domain.

Syntax

mkcomg [**-s** *sensitivity*] [**-p** *period*] [**-g** *grace*] [**-t** *priority*] [**-x b** | **r** | **br**] [**-N** *UseForNodeMembership*] [**-e** *NIM_path*] [**-m** *NIM_parameters*] [**-M** *media_type*] [**-i** {**h** | **n**}**:***interface1*[**:***node1*][**,***interface2*[**:***node2*]...] | **-S** {**h** | **n**}**:"***interface_selection_string***"** [**-6**] [**-h**] [**-TV**] *communication_group*

Description

The **mkcomg** command creates a new communication group definition for an online peer domain with the name specified by the *communication_group* parameter. The communication group is used to define heartbeat rings for use by topology services and to define the tunables for each heartbeat ring. The

communication group determines which devices are used for heartbeating in the peer domain. There can be more than one communication group in a peer domain.

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

The **-e** and **-m** flags are used to set the network interface module (NIM) path and parameters. The NIM path is the path to the NIM that supports the adapter types used in the communication group. The NIM parameters are passed to NIM when it is started. If **-m** is not specified, the parameters predefined by topology services are used.

The communication group can be assigned to one or more interface resources. Use the **-i** flag to assign the communication group to a specific interface resource name. The interface resource can be limited to one on a particular node. An interface resource can also be specified using the **-S** flag and a selection string. This is used when specifying the interface resource name is not sufficient. The **-i** and **-S** flags cannot be used together. The **chcomg** command can also be used to assign a communication group to an interface resource.

Flags

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

-p *period*
: Specifies the amount of time between heartbeats. The period is specified in seconds and is significant to milliseconds. It can be specified as an integer or as a floating-point number.

-g *grace*
: Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

  The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

  0 Specifies that the grace period is disabled.

  -1 | D Specifies that the topology services subsystem controls the grace period. This is the default.

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

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

  b Specifies that the broadcast feature should not be used even if the underlying media support it. If **-x b** is not specified, the broadcast feature will be used if the underlying media support it.

  r Specifies that the source routing feature should not be used even if the underlying media support it. If **-x r** is not specified, the source routing feature will be used if the underlying media support it.

To exclude more than one control, specify the feature characters consecutively: **-x br**.

-N *UseForNodeMembership*
Specifies whether group services will use the communication group in calculating node membership. Sets the **UseForNodeMembership** persistent resource attribute for the communication group resource. Valid values are:

0 Indicates that, regardless of the results of liveness checks run on **NetworkInterface** resources that are members of this communication group, group services will not use those results in calculating whether the node owning the interfaces is online.

1 Indicates that group services will use the results of liveness checks run on the **NetworkInterface** resources in calculating the online state of their owning nodes.

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

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

-M *media_type*
Specifies the type of interfaces that make up *communication_group*. Valid values are:

0 Indicates that *communication_group* consists of interface resources other than IP or disk.

1 Indicates that *communication_group* consists of IPv4 or IPv6 interface resources.

If the **-M** flag is not specified, this is the default.

2 Indicates that *communication_group* consists of disk interface resources.

-i {h | n}:*interface1***[:***node1***] [,***interface2***[:***node2***]...**
Assigns *communication_group* to one or more heartbeat or network interface resources and, optionally, to the nodes where these resources can be found. Specify **-i h** for heartbeat interface resources or **-i n** for network interface resources.

By default, the **-i n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-i n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

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

-S {h | n}:"*network_selection_string***"**
Assigns *communication_group* to the heartbeat or network interface that is specified by *interface_selection_string*. Specify **-S h** for heartbeat interfaces or **-S n** for network interfaces.

By default, the **-S n** flag adds network interface resources that have IPv4 addresses to *communication_group*. If the **-6** flag is specified, the **-S n** flag will add network interface resources that have IPv6 addresses to *communication_group*.

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

-6 Specifies that IPv6 addresses represented as resources on each interface have their communication group changed to the one specified. IPv4 addresses represented as resources on the interfaces would be unaffected.

By default (without **-6** specified), the inverse is true. Only IPv4 addresses represented as resources on the interface would have their communication group changed.

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

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

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

Parameters

communication_group
Specifies the name of the new communication group that is to be created for the online peer domain. The name can contain any printable character.

Security

The user of the **mkcomg** command needs write permission for the **IBM.CommunicationGroup** resource class. Write permission for the **IBM.NetworkInterface** resource class is required to set the communication group for a network interface resource. By default, **root** on any node in the peer domain has read and write access to these resource classes through the configuration resource manager.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be defined.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

   ```
   mkcomg ComGrp1
   ```

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

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

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

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

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

   ```
   mkcomg -x br ComGrp1
   ```

5. To define the communication group **ComGrp1** for the peer domain **ApplDomain**, using a NIM path of **/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 **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource named **hbi0** on **nodeC**, run this command on **nodeA**:

   ```
   mkcomg -i h:hbi0:nodeC ComGrp1
   ```

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

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

8. To define the communication group **ComGrp1** for **ApplDomain** and assign **ComGrp1** to the heartbeat interface resource that uses the subnet 9.345.67.812, run this command on **nodeA**:

   ```
   mkcomg -S h:"Subnet == 9.345.67.812" ComGrp1
   ```

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

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

10. To define the communication group **ComGrp1** for **ApplDomain**, using a period of 500 milliseconds, run this command on **nodeA**:

    ```
    mkcomg -p 0.5 ComGrp1
    ```

Location

/usr/sbin/rsct/bin/mkcomg

mkrpdomain Command

Purpose

Creates a peer domain definition.

Syntax

To create a peer domain definition, by:

- Specifying node names on the command line:

  mkrpdomain [**-t** *TS_port*] [**-g** *GS_port*] [**-Q** *quorum_type | quorum_type_name*] [**-c**] [**-m** *fanout*] [**-k** *cssk_type* [**-r** *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [**-h**] [**-TV**] *peer_domain node_name1* [*node_name2 ...*]

- Using a list of node names in an input file:

  mkrpdomain -f | **-F** { *file_name* | **"–"** } [**-t** *TS_port*] [**-g** *GS_port*] [**-Q** {*quorum_type | quorum_type_name*}] [**-c**] [**-m** *fanout*] [**-k** *cssk_type* [**-r** *refresh_interval*]] [**–6**] [**-C** *cluster_type* **-R** *repository_disk* [**-D** *shared_disk1*[**,***shared_disk2*...]]] [**-h**] [**-TV**] *peer_domain*

| To create a peer domain definition with the policy information:

mkrpdomain [**-p** *Policy*] *ApplDomain nameA* [*@host_nameA*] [*nameB* [*@host_nameB*] ...]

Description

The **mkrpdomain** command creates a peer domain definition with the name specified by the *peer_domain* parameter. The nodes that are specified by *node_name* are defined to the new peer domain. A peer domain can be used to provide high-availability services when you configure application and system resources.

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

The **mkrpdomain** command fails if one or more of these situations occurs:

- The name of the peer domain is already in use.
- One or more nodes cannot be successfully defined to the peer domain.
- The UDP port numbers for group services and topology services are not available on all of the nodes to be defined to the peer domain.

Use the **-c** flag to enable **mkrpdomain** to continue when there is an error on one of the nodes. The peer domain quorum rules can be modified by using the **-Q** flag. The quorum rules determine under what conditions operational changes, such as starting or stopping resources, and configuration changes, such as adding or removing a node, can be made. Start up quorum defines how many nodes are contacted to get configuration information to start the peer domain. In a typical environment, two quorum rule types are used: normal and quick. For the quick quorum type, only one node is contacted before the peer domain group is started. Operational and configuration quorum rules are the same. To see what quorum rule types are available on a node, run:

```
lsrsrc -c IBM.PeerDomain AvailableQuorumTypes
```

You can use the **-k** flag to set the cluster shared secret key (CSSK). The CSSK is used for message authentication in the peer domain. By default, the CSSK is disabled (that is, set to CSSKTYPE_None). To enable message authentication, use a CSSK value such as CSSKTYPE_DES_MD5 with the **-k** flag. Enabling message authentication affects performance. The complexity of the encryption algorithm determines the effect.

Message authentication also requires that the time-of-day clocks (TODs) of the nodes in the peer domain are synchronized — according to the system time — to within 2 minutes of each other. When the nodes' TODs are synchronized across the peer domain, this function helps to defend against message replay attacks. If the nodes' TODs are not synchronized to within 2 minutes of each other, messages that are passed between a sending node and a receiving node with a time difference that is longer than 2 minutes are discarded.

When message authentication is enabled by using the **-k** flag, a key refresh interval can be specified by using the **-r** flag. By default, the key is refreshed daily.

To change the CSSK type for a peer domain, use the **chrsrc** command. For example:

```
chrsrc -c IBM.RSCTParameters CSSKType=cssk_type
```

To list the CSSK type that is used for an online peer domain, use the **lsrsrc** command. For example:

```
lsrsrc -c IBM.RSCTParameters CSSKType
```

To cause the CSSK to be refreshed, use the **runact** command. For example:

```
runact -c IBM.PeerDomain UpdateKey
```

For information about setting up and managing CSSK settings, see the *Administering RSCT* guide.

Use the **-6** flag to establish a peer domain in which the IPv6 addresses that are configured on the nodes' network interfaces are visible as resources in **IBM.NetworkInterface** class. These IPv6 addresses are not used for heartbeating or internal peer domain operations. If the -6 flag is not specified, no IPv6 addresses are visible as resources in **IBM.NetworkInterface**.

The **mkrpdomain** command does not bring the peer domain online automatically. To bring the peer domain online, run the **startrpdomain** command. You can add nodes to the peer domain by using the **addrpnode** command. To remove nodes from the peer domain, use the **rmrpnode** command.

A node can be defined in more than one peer domain but it can be online in only one peer domain at a time.

Flags

Item	Description
-6	Specifies that the IPv6Support persistent class attribute of the **IBM.NetworkInterface** class has a value of 1 rather than the default (0) in the peer domain that is to be created. For any IP interface on any node in a cluster that has one or more IPv6 addresses configured, only one of these IPv6 addresses are made visible as a resource in **IBM.NetworkInterface**. Therefore, if a network interface has IPv4 addresses and IPv6 addresses configured on it, two resources in **IBM.NetworkInterface** refers to the interface (through the Name attribute), one with the IP address value set to the primary IPv4 address, and one with the selected IPv6 address. If multiple IPv6 addresses are configured on an interface, preference is given to global addresses over link-local addresses for representation as a resource. In addition, IPv6 addresses are used for heartbeating or internal peer domain operations. **Note:** Even if IPv6Support is changed, the current registered applications do not receive the notification for any resource addition or deletion until the domain or the **IBM.ConfigRM** class is restarted.

Item	Description
-c	Continues to run the **mkrpdomain** command on the remaining nodes. By default, if the **mkrpdomain** command fails on any node, it fails on all nodes. The **-c** flag overrides this behavior, so that the **mkrpdomain** command runs on the other nodes, even if it fails on one node.
-C *cluster_type*	Specifies the cluster type. Valid values are as follows: **0** Creates a peer domain. This value is the default. **1** Creates a peer domain and the underlying Cluster-Aware AIX (CAA) cluster. If you specify the **-C 1** flag, you must also specify a repository disk by using the **-R** flag. Also, you can optionally specify one or more shared disks by using the **-D** flag.
-D *shared_disk1* [,*shared_disk2*...]	Specifies one or more shared disks for a CAA cluster. If you specify the **-D** flag, you must also specify the **-C** and **-R** flags.

Item

-f | **-F** { *file_name* | **"-"**

Description

Specifies that node names are read from a file or from standard input. Use **-f** *node_file* or **-F** *node_file* to read the node names from a file.

Note: The command requires that the following conditions be met to display a valid output:

- Specify 1 node name per line. The command ignores any blank characters to the left of the node name.
- Use a number sign (#) to indicate that the remainder of the line (or the entire line if the # is in column 1) is a comment.
- Specify the actual host name of the node by using @ sign without any space between node name and its host name. An example of the syntax follows:

  ```
  [nodeA@hostA]
  ```

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

- are group services group leader candidates
- are used for quorum decisions
- have access to the peer domain's tiebreaker mechanism

You can customize node characteristics by using an at sign (@) control character followed by one or more of these special characters:

P | p Specifies that the node is a group services group leader candidate.

Q | q Specifies that the node is a quorum node.

B | b Specifies that the node has access to the peer domain's tiebreaker mechanism. B or b can be specified for quorum nodes only.

! Specifies that the node does not have a certain characteristic. For example, **!Q** indicates that the node is not a quorum node.

When customizing node characteristics, consider the following (where x is P, Q, or B):

- Use only one @ control character per line, followed immediately by one or more special characters, after the node name and before any comments.
- Do not specify !QB for a node, as it results in an error.
- If you use a node number, add it after the node name and before any comments. The node number can precede or follow the node characteristic specifications.
- If x is specified for one or more nodes and !x is not specified for any nodes, the nodes that do not have an x specified are assumed to have a value of !x.
- If !x is specified for one or more nodes and x is not specified for any nodes, the nodes that do not have an !x specified are assumed to have a value of x.
- If x and !x are specified for different nodes in the same node file, all of the nodes in the file must have a specification of x or !x.

See the *Administering RSCT* for more information.

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

-g *GS_port*

Specifies the group services port number. This UDP port is for daemon-to-daemon communication. Any unused port in the range 1024 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12348.

-h

Writes the command's usage statement to standard output.

Item	Description
-k *cssk_type*	Specifies the cluster shared secret key (CSSK) to be used for message authentication in the peer domain. Use the CSSK that best suits your applications in terms of the degree of data protection, overhead, and performance. The longer the key, the stronger the encryption algorithm. The stronger the algorithm, the slower the performance. The valid key types are as follows: **CSSKTYPE_None** Indicates that message authentication is disabled. This value 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. You must be running RSCT 2.4.7.1 or later to use this flag.
-m *fanout*	Specifies the maximum number of threads to use in parallel operations for the specified peer domain. This value is stored as a persistent attribute in the peer domain's **IBM.PeerNode** class. *fanout* can be an integer from **16** to **2048**. If this flag is not specified, the default value (**128**) is used.
-p *Policy*	Reads the policy from the user input when the **mkrpdomain** command creates the domain. You can use this command to specify the policy information when you create the domain. The valid values for the *Policy* attribute are 0 and 1. If you do not specify the **-p** flag for the **mkrpdomain** command, the default value 0 is set in non-CAA clusters and 1 is set in CAA clusters. If the value of policy is set as 1, the **Name** field of the **IBM.PeerNode** class is maintained in sync with the host name of the **IBM.PeerNode** class. If the value of policy is set as 0, the **Name** field is not maintained in sync with the host name, irrespective of the domain. However, the **-p** *0* flag cannot be specified for CAA domain as a limitation. The policy information can be changed by using a **chrsrc** class action after the cluster is created.

Item	Description
-Q *quorum_type* | *quorum_type_name*	Specifies the quorum rules that are used for startup, operational, and configuration quorum. Startup quorum defines how many nodes are contacted to obtain configuration information before the peer domain is started. Operational quorum defines how many nodes must be online to start and stop resources and how tie breaking is used. Configuration quorum defines how many nodes must be online to change the peer domain (adding or removing a node, for example). To see what quorum rule types are available on a node, run: `lsrsrc -c IBM.PeerDomain AvailableQuorumTypes` The valid values are as follows: **0 | normal** Specifies normal quorum rules. This value is the default. For startup quorum, at least half of the nodes are contacted for configuration information. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum. **1 | quick** Specifies quick quorum rules. For startup quorum, even if no other nodes can be contacted, the node still comes online. For configuration quorum, more than half of the nodes must be online to make configuration changes. For operational quorum, the cluster or subcluster must have a majority of the nodes in the peer domain. If a tie exists between subclusters, the subcluster that holds the tiebreaker has operational quorum.
-r *refresh_interval*	Specifies the CSSK refresh interval when message authentication is enabled in the peer domain. This is the interval at which the CSSK is refreshed. The format of *refresh_interval* is: *dd***:***hh***:***mm***:***ss*, where *dd* is the number of days between key refreshes, *hh* is the number of hours, *mm* is the number of minutes, and *ss* is the number of seconds. The *refresh_interval* value can be truncated on the right, so **-r 5** means refresh every 5 days and **-r 0:12** means refresh every 12 hours. The default refresh interval is 1 day. The minimum refresh interval is 30 seconds. The maximum refresh interval is 30 days. The **-r** flag can be specified when the **-k** flag is used. You must be running RSCT 2.4.7.1 or later to use this flag.
-R *repository_disk*	Specifies the repository disk for a CAA cluster. If you specify the **-R** flag, you must also specify the **-C** flag.
-t *TS_port*	Specifies the topology services port number. This UDP port is used for daemon-to-daemon communication. Any unused port in the range 1024 - 65535 can be assigned. The command fails if the specified port is unavailable. The default is 12347.
-T	Writes the command's trace messages to standard error. For your software service organization's use only.
-V	Writes the command's verbose messages to standard output.

Parameters

peer_domain

Specifies the name of the new peer domain to be created. You can use these ASCII characters only in the peer domain name: **A** to **Z**, **a** to **z**, **0** to **9**, **.** (period), and _ (underscore). In addition, the peer domain name *cannot* be **IW**.

node_name1 **[***node_name2* **...]**
Specifies the node (or nodes) to include in this peer domain definition. The node name is the IP address or the long or short version of the DNS host name. The node name must resolve to an IP address.

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the Resource Monitoring and Control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

Any node to be defined to the peer domain must be reachable from the node on which this command runs.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

   ```
   mkrpdomain ApplDomain nodeA
   ```

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

   ```
   mkrpdomain ApplDomain nodeA nodeB nodeC
   ```

3. To define a peer domain that is called **ApplDomain** that consists of 2 nodes that are called **nodeA** and **nodeB**, with a topology services port number of 1200 and a group services port number of 2400, run this command on **nodeA** or **nodeB**:

   ```
   mkrpdomain -t 1200 -g 2400 ApplDomain nodeA nodeB
   ```

4. To define a peer domain that is called **ApplDomain** that consists of 2 nodes that are called **nodeA** and **nodeB** by using message authentication key algorithm **CSSKTYPE_DES_MD5**, run this command on **nodeA** or **nodeB**:

   ```
   mkrpdomain -k CSSKTYPE_DES_MD5 ApplDomain nodeA nodeB
   ```

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

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

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

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

6. To define a peer domain that is called **ApplDomain** that consists of **nodeA**, **nodeB**, **nodeC**, **nodeD**, and **nodeE**, by using the **/pd/pdnodes.config** file, which specifies that **nodeA** has access to the peer domain's tiebreaker mechanism, **nodeB** and **nodeC** cannot be used in quorum decisions, and **nodeC** and **nodeD** cannot be the group services group leader, run the following command on any of the nodes:

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

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

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

7. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, with the policy **NamePolicy 1**, run the following command:

   ```
   mkrpdomain -p 1 ApplDomain nodeA nodeB
   ```

NamePolicy 1 means that any change in host name also updates the node name. In this case, the host name is not specified in the beginning. Hence, the node names (**nodeA** and **nodeB**) are set as host names for the respective nodes.

8. To define a peer domain that is called **ApplDomain**, which consists of 2 nodes that are called **nodeA** and **nodeB**, whose host names are **hostA** and **hostB**, run the following command:

```
mkrpdomain ApplDomain nodeA@hostA nodeB@hostB
```

These host names are the actual host names that are used for communication.

Location

/usr/sbin/rsct/bin/mkrpdomain

Files

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

preprpnode Command

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.

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.

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.

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 the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

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.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

   ```
   preprpnode nodeA
   ```

2. Suppose **nodeA** and **nodeB** are online in **ApplDomain**. 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
   ```

Location

/usr/sbin/rsct/bin/preprpnode

Files

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

rmcomg Command

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 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.

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.

Parameters

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

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.

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.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is defined and online to the peer domain where the communication group is to be removed.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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

```
rmcomg ComGrp1
```

Location

/usr/sbin/rsct/bin/rmcomg

rmrpdomain Command

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 subcluster. Later, the majority subcluster may become active. If so, the domain is still removed.

If a Cluster-Aware AIX (CAA) cluster is configured and this peer domain is representing it, the **rmrpdomain** command removes the underlying CAA cluster as well.

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.

Parameters

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

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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the

RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

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 **rsct.basic.rte** fileset for AIX®.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
   ```

Location

/usr/sbin/rsct/bin/rmrpdomain

Files

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

rmrpnode Command

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.

If a Cluster-Aware AIX (CAA) cluster is configured and this peer domain is representing it, the **rmrpnode** command removes the nodes from the underlying CAA cluster as well.

Specifying the **-f** flag forces the specified nodes to be removed from the peer domain. When the last tiebreaker node is removed using **rmrpnode -f**, only the remaining quorum nodes (as opposed to all nodes) are converted to being tiebreaker nodes.

If the **-f** flag is not specified when this command is run:

- more than half of the quorum nodes must be online to remove one or more nodes from the domain
- an error is returned if the peer domain has no remaining tiebreaker nodes as a result

See the *Administering RSCT* for more information about quorum nodes and tiebreaker nodes.

Flags

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

When the last tiebreaker node is removed using this flag, only the remaining quorum nodes (as opposed to all nodes) are converted to being tiebreaker nodes.

See the *Administering RSCT* for more information about quorum nodes and tiebreaker nodes.

-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.

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.

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.

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.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT

When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain 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 **rsct.basic.rte** fileset for AIX®.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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
```

Location

/usr/sbin/rsct/bin/rmrpnode

startrpdomain Command

Purpose

Brings a peer domain that has already been defined online.

Syntax

startrpdomain [**-A** | **-L**] [**-t** *timeout*] [**-Q** *quorum_type* | *quorum_type_name*] [**-m** *fanout*] [**-h**] [**-w** [-s *Seconds*]] [**-TV**] *peer_domain*

Description

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

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

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

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

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

Flags

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

-L Uses the latest version of the peer domain configuration information that is on the local node. This flag cannot be specified if the **-A** flag is specified. If neither flag (**-A** or **-L**) is specified, the latest version of the peer domain configuration information from at least half of the nodes in the peer domain is used.

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

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

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

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

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

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

-s Specifies the wait time in seconds for the peer domain to be online before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the online operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until the peer domain is online (no timeout on waiting).

-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.

-w Waits for the peer domain to be online before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

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

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

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

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** is one of the nodes defined to **ApplDomain**.

1. To bring **ApplDomain** online, run this command on **nodeA**:
   ```
   startrpdomain ApplDomain
   ```
2. To bring **ApplDomain** online using all of the nodes in the peer domain to obtain the latest version of the peer domain configuration information, run this command on **nodeA**:
   ```
   startrpdomain -A ApplDomain
   ```
3. To bring **ApplDomain** online using a peer domain configuration timeout value of 240 seconds (to make sure that at least half of the nodes in the peer domain are used), run this command on **nodeA**:
   ```
   startrpdomain -t 240 ApplDomain
   ```

Location

/usr/sbin/rsct/bin/startrpdomain

startrpnode Command

Purpose

Brings one or more nodes online to a peer domain.

Syntax

startrpnode [**-h**] [**-w** [-s *Seconds*]] [**-TV**] *node_name1* [*node_name2* ...]

startrpnode -f | **-F** { *file_name* | **"–"** } [**-h**] [**-w** [-s *Seconds*]] [**-TV**]

Description

The **startrpnode** command brings one or more offline nodes online to a peer domain. The peer domain is determined by the online peer domain where the command is run. The command must be run from a node that is online to the desired peer domain.

The node that is being brought online must have already been defined to be in this peer domain using the **addrpnode** command or the **mkrpdomain** command. The node must not be online to any other peer domain.

Flags

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

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

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

-s
: Specifies the wait time in seconds for all of the specified nodes to be online before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the online operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until all of the specified nodes are online (no timeout on waiting).

-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.

-w Waits for all of the specified nodes to be online before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

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

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run from a node that is online to the peer domain. The node that is to be brought online must be offline to the peer domain, must not be online to any other peer domain, and must be reachable from where the command is run.

Implementation Specifics

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

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

In this example, **nodeA** is defined and online to **ApplDomain**, **nodeB** is reachable from **nodeA**, and **nodeB** is not online to **ApplDomain** or any other peer domain. To bring **nodeB** online to **ApplDomain**, run this command from **nodeA**:

```
startrpnode nodeB
```

Location

/usr/sbin/rsct/bin/startrpnode

stoprpdomain Command

Purpose

Takes an online peer domain offline.

Syntax

| **stoprpdomain** [**-f**] [**-h**] [**-w** [-s *Seconds*]] [**-TV**] *peer_domain*

Description

The **stoprpdomain** command takes all of the nodes that are currently online in the peer domain offline. The peer domain definition is not removed from the nodes.

The command must be run on a node that is online in the peer domain. If the command is run on a node that is offline to the peer domain, no action is performed.

If a Cluster-Aware AIX (CAA) cluster is configured, no action is performed because a peer domain operation in a CAA environment exists and is online for the life of the CAA cluster.

The **-f** flag must be used to override a subsystems rejection of the request to take the peer domain offline. A subsystem may reject the request if a peer domain resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystems that the peer domain must be brought offline regardless of the resource state.

Flags

-f Forces the subsystems to accept the stop request when it otherwise would not.

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

| **-s** Specifies the wait time in seconds for the peer domain to be offline before the command

completes when the **-s** flag is used with the **-w** flag . If the waiting time exceeds the number of seconds, the command returns, but the offline operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until the peer domain is offline (no timeout on waiting).

-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.

-w Waits for the peer domain to be offline before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

peer_domain
Specifies the name of the online peer domain that is to be brought offline.

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

6 The peer domain definition does not exist.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online in the peer domain.

Implementation Specifics

This command is part of the **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** is one of the nodes defined and is online to **ApplDomain**.

1. To take **ApplDomain** offline, run this command on **nodeA**:
   ```
   stoprpdomain ApplDomain
   ```
2. To take **ApplDomain** offline while making sure the stop request will not be rejected by any subsystem, run this command on **nodeA**:
   ```
   stoprpdomain -f ApplDomain
   ```

Location

/usr/sbin/rsct/bin/stoprpdomain

stoprpnode Command

Purpose

Takes one or more nodes offline from a peer domain.

Syntax

stoprpnode [**-f**] [**-h**] [**-w** [-s *Seconds*]] [**-TV**] *node_name1* [*node_name2*...]

stoprpnode -F { *file_name* | **"–"** } [**-f**] [**-h**] [**-w** [-s *Seconds*]] [**-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.

If a Cluster-Aware AIX (CAA) cluster is configured, no action is performed because a peer domain operation in a CAA environment exists and is online for the life of the CAA cluster.

The **-f** flag must be used to override a subsystem's rejection of the request to take a node offline. A subsystem may reject the request if a node resource is busy, such as in the case of a shared disk. Specifying the **-f** flag in this situation indicates to the subsystems that the node must be brought offline regardless of the resource state.

If this command is used to take more than one node offline by specifying more than one *node_name* parameter, and the node that this command is running on is in the list, it will be brought offline last.

Flags

-f Forces the subsystems to accept the stop request when it otherwise would not.

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

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

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

-s Specifies the wait time in seconds for all of the specified nodes to be offline before the command completes when the **-s** flag is used with the **-w** flag. If the waiting time exceeds the number of seconds, the command returns, but the offline operation continues. The default value is 300 seconds (5 minutes). Use 0 to specify that the command must not return until all of the specified nodes are offline (no timeout on waiting).

-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.

-w Waits for all of the specified nodes to be offline before the command completes. Use the **-s** flag to specify the waiting time in seconds.

Parameters

node_name1 **[***node_name2***...]**
Specifies the peer domain node names of the nodes that are to be brought offline from the peer domain. You must specify the node names in exactly the same format as they were specified with the **addrpnode** command or the **mkrpdomain** command. To list the peer domain node names, run the **lsrpnode** command.

Security

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

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the

RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

Restrictions

This command must be run on a node that is online 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 **rsct.basic.rte** fileset for the AIX® operating system.

Standard Input

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

In these examples, **nodeA** and **nodeB** are online to **ApplDomain**.

1. To take **nodeB** offline, run this command on **nodeA**:
   ```
   stoprpnode nodeB
   ```
2. To take **nodeB** offline and force the offline request, run this command on **nodeA**:
   ```
   stoprpnode -f nodeB
   ```

Location

/usr/sbin/rsct/bin/stoprpnode

Cluster configuration commands

This section describes the **ctadmingroup** command, various cluster configuration files, and Common Information Model (CIM) resource manager commands. You can define a cluster administration group by using the **ctadmingroup** command. CIM is a data model, similar to the RMC data model, for organizing computer hardware and software resources into a common object-oriented class hierarchy.

ctadmingroup Command

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, which are created after the **ctadmingroup** command is run, contain the new permissions. 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 the **ctadmingroup** command with:

- a different group name, the new group that is specified becomes the cluster administration group, thus replacing the previous group.
- no flagsoptions 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** flagoption, it removes the cluster administration group. After the group is removed, users who belong to that group might 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 trace file of the security subsystem is configurable. To determine the location of the trace file, the **ctadmingroup** command requests information from 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 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 might 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 usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

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 trace file of the security subsystem.

/usr/sbin/rsct/cfg/ctcasd.cfg
The secondary location of the cluster security configuration file. The **ctadmingroup** command requests information from 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 flagoption 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** flagoption is specified, this command 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 can 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 AIXpackage for Linux.

Location

/usr/sbin/rsct/bin/ctadmingroup

Examples

1.

To display the group name and ID of the cluster administration group, enter:

```
ctadmingroup
```

Configuration files

ctcas_hba2.map File
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 support 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 must 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 must not be modified directly. Instead, the file must be copied to **/var/ct/cfg/ctcas_hba2.map**, and modifications must be made to this copy. The default configuration file must 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, it can 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 run 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 must not be modified unless the administrator is instructed to do so by the cluster software service provider. Incorrect modification of this file results 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 must not be modified directly. Instead, the file must be copied to **/var/ct/cfg/ctcas_hba2.map**, and modifications must be made to this copy. The default configuration file must 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
```

ct_class_ids File
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 file 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 File
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 must 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**

ctfile.cfg File
Purpose

Controls tracing and logging for various RSCT components.

Description

To initiate tracing and logging control, system administrators can create the RSCT file configuration (RFC) file, **/etc/ctfile.cfg**. A system administrator can copy a sample configuration file, **/usr/sbin/rsct/cfg/ctfile.cfg**, to the **/etc** directory, and then modify it. The control information in the configuration file takes effect the next time a specified component, a daemon, for example, is started.

RSCT components that support tracing and logging control, which are listed in the sample configuration file, obtain the following information from **/etc/ctfile.cfg**:

- An absolute path to a directory tree under which the component trace files and log files are to be written. This path can refer to a remote file system.
- A flag that indicates whether tracing is to be disabled for each component.
- A flag that indicates whether tracing is to be disabled for all RSCT components.
- When tracing is enabled, the size of a trace file for each component and each trace file. Some components write to several trace files.
- The amount of information to be traced.
- A flag that indicates whether logging is to be disabled for each component.
- A flag that indicates whether logging is to be disabled for all RSCT components.

Lines that consist of only the characters **NL** or white space are ignored. If the first non-white-space character of a line is the number sign character (**#**), the line is a comment, so it is ignored. All other lines must contain one or more tokens. If there is more than one token on a line, the tokens must be separated by white space. A token line can contain leading or trailing white space. White space is any mix of the blank and tab characters.

The trace and log root directory token

The trace and log root directory token is a name/value pair. The format is:

```
CT_TRACE_LOG_ROOT_DIR=path_name
```

The path name must be absolute. If this token is not specified, the trace and log root directory is assumed to be **/var/ct**. In either case, all trace and log file path names are constructed with the trace and log root directory as the path name prefix.

The trace disable token

The trace disable token is a name/value pair followed by an optional component name. The format is:

```
CT_TRACE_DISABLE={true|false} [component_name]
```

This token can be specified more than once, each on a separate line. If this token is not specified then tracing is enabled for all components that reference this file and are not otherwise specified in another

trace disable token line. If this token is specified with a value of true, and no component name is specified, tracing is disabled for all components that reference this file and are not otherwise specified in another trace disable token line. If this token is specified with a value of false, and no component name is specified, then tracing is enabled for all components that reference this file and are not otherwise specified in another trace disable token line. If a component name is specified then tracing for that component is enabled or disabled to match the value false or true, respectively. For a component, the token line that specifies that component has precedence over any token line that does not specify a component. If more than one token line specifies the same component, the last such token line has precedence. Component names are the resource manager names in the case of resource managers (as derived from the resource manager **.mdef** file), **mc** in the case of the RMC daemon and **ctcasd** in the case of the cluster authentication services daemon. Specifically, the component name must be the name used in constructing the full path name of a trace file. Here is an example:

```
CT_TRACE_DISABLE=true
CT_TRACE_DISABLE=false mc
CT_TRACE_DISABLE=false IBM.MgmtDomainRM
```

Tracing is enabled for the management domain resource manager and the RMC daemon. For all other components trace is disabled.

The trace file size token

The trace file size token is a name/value pair followed by a component name and trace file name. It has the following form:

```
CT_TRACE_FILE_SIZE=nnn[K] component_name file_name
```

where *nnn* is the size of the specified trace file in bytes. If the optional **K** suffix is specified, the size is specified in units of 1024 bytes. The RSCT trace facility rounds up file sizes to a page boundary. If a trace file size token is not specified, components use the file sizes that are programmed into the component.

The trace level token

The trace level token is a name/value pair followed by a component name. The format is:

```
CT_TRACE_LEVELS=string component_name
```

The level string is in standard trace facility format, for example, Comp_ID:category=level. If a trace level token is not specified, components use the trace levels programmed into the component. Trace levels determine the amount of information recorded in a trace file.

The logging disable token

The logging disable token is a name/value pair followed by an optional component name. The format is:

```
CT_LOGGING_DISABLE={true|false} [component_name]
```

This token can be specified more than once, each on a separate line. If this token is not specified, logging is enabled for all components that refer to this file and are not otherwise specified in another logging disable token line. If this token is specified with a value of **true** and no component name is specified, logging is disabled for all components that refer to this file and are not otherwise specified in another logging disable token line. If this token is specified with a value of **false** and no component name is specified, logging is enabled for all components that refer to this file and are not otherwise specified in another logging disable token line. If a component name is specified, logging for that component is enabled or disabled to match the value **false** or **true**, respectively. For a component, the token line that specifies that component has precedence over any token line that does not specify a component. If more than one token line specifies the same component, the last such token line has precedence. Component names are the resource manager names in the case of resource managers (as derived from the resource manager's **.mdef** file), **mc** in the case of the RMC daemon and **ctcasd** in the case of the cluster

authentication services daemon. Specifically, the component name must be the name that is used in constructing the full path name of a logging file.

Files

/etc/ctfile.cfg
Location of the RSCT file configuration (RFC) file that is created by the system administrator

/usr/sbin/rsct/cfg/ctfile.cfg
Location of the sample **ctfile.cfg** file

Implementation specifics

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

Location

/etc/ctfile.cfg

ctgroups File
Purpose

Contains the group name of the cluster administration group.

Description

The **ctgroups** file stores the group name of the cluster administration group. In addition, **ctgroups** caches the corresponding group ID.

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/ctgroups
Contains the **ctgroups** file

ct_has.pkf File
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 will expect 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 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 (or IP addresses) associated with all AF_INET-configured adapters that the daemon can detect. This may cause inadvertent authentication failures 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 UNIX-identity-based authentication 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 UNIX-identity-based authentication.

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.

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
 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:Perf=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
SERVICES= hba CAS
```

Location

/var/ct/cfg/ct_has.pkf
Contains the **ct_has.pkf** file

Files

/usr/sbin/rsct/cfg/ctcasd.cfg
Default location of the **ctcasd.cfg** file

ct_has.qkf File
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** demon 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 (or IP addresses) associated with all AF_INET-configured adapters that the daemon can detect. This may cause inadvertent authentication failures 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 UNIX-identity-based authentication 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 not configured accurately 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.

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
 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:Perf=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
 SERVICES= hba CAS
```

Location

/usr/sbin/rsct/bin/ct_has.qkf
: Location of the **ct_has.qkf** file.

Files

/usr/sbin/rsct/cfg/ctcasd.cfg
: Default location of the **ctcasd.cfg** file

ct_has.thl File
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 in order to decrypt UNIX-identity-based 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 UNIX-identity-based authentication.

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 AF_INET-configured adapters that the daemon can detect. This may cause inadvertent authentication failures if the public and private key files were accidentally or intentionally removed from the local system before the daemon was restarted. The **ctcasd** daemon creates new keys for the node, which will not match the keys stored on the other cluster nodes. If UNIX-identity-based authentication 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 UNIX-identity-based authentication.

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.

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
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:Perf=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
 SERVICES= hba CAS
```

Location

/usr/sbin/rsct/bin/ct_has.thl
Location of the **ct_has.thl** file.

Files

/usr/sbin/rsct/cfg/ctcasd.cfg
Default location of the **ctcasd.cfg** file

ctcasd.cfg File
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. Acceptable values are **ON** and **OFF**. If the **TRACE**

attribute is not listed in the **ctcasd.cfg** file, tracing is not activated. 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.

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. Possible values are: **1**, **2**, **4**, and **8**.

_SEC:API
Tracks the entry and exit of subroutines within the daemon. Possible values are: **1** and **8**.

_SEC:Perf
Captures performance-related information. Possible values are: **1**, **4**, and **8**.

_SEC:Info
Traces the general execution progress of the daemon. Possible values are: **1**, **2**, **3**, **4**, and **7**.

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 performance data and highly-detailed error data, the specification for TRACELEVELS would be:

```
TRACELEVELS=_SEC:Perf=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 **rsa1024** is used.

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.

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.

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
 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:Perf=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
 SERVICES= hba CAS
```

Location

/var/ct/cfg/ctcasd.cfg
Contains the **ctcasd.cfg** file

Files

/usr/sbin/rsct/cfg/ctcasd.cfg
Default location of the **ctcasd.cfg** file

ctrmc.acls File
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 directory **/usr/sbin/rsct/cfg**. You can allow RMC to use the default permissions set in this file, or you can modify the file after copying it to the directory **/var/ct/cfg/**. For more information, see the *RSCT: Administration Guide*.

For information about how access controls are implemented for the **IBM.LPCommands** resource class and its resources, see the **lpacl** information file.

Implementation Specifics

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

Location

/var/ct/cfg/ctrmc.acls
: Contains the **ctrmc.acls** file

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

ctrmc.rio File
Purpose

Adjusts reporting intervals for resource dynamic attributes.

Description

To adjust the reporting intervals for resource dynamic attributes that are defined in a class definition file, you can create a *reporting interval override file* called **ctrmc.rio**. A *reporting interval* is the amount of time between a resource manager's sampling of values. You can set reporting intervals for all classes, a single class, or a single attribute of a class. Information in the **ctrmc.rio** file applies only to resource dynamic attributes that have reporting intervals, which have a variable type of **Counter** or **Quantity**.

The **ctrmc.rio** file is a plain text file, in which a line in the file contains two white-space-separated tokens, as follows:

class_name[:*attr_name*] *reporting_interval*

The first token is a class name as returned by the **lsrsrc** command, or, a class name followed immediately by a colon, which is followed by an optional resource dynamic attribute name as returned by the **lsrsrcdef** command. If only a class name is specified, the reporting interval applies to all resource dynamic attributes of the class that have reporting intervals. If a class name and attribute name are specified, the reporting interval applies to the named attribute only, if appropriate. If the attribute does not have a reporting interval, the line is ignored. The second token, the reporting interval, is an unsigned

integral value that is interpreted as a number of seconds. If the class name, without any attribute names, is the keyword **ALL**, the reporting interval applies to all dynamic resource attributes of all resource classes. The last specification for an attribute applies.

In the following example:

```
Foo:larry       10
Foo             15
Foo:curly       20
```

the reporting interval would be **15** for **larry** and **20** for **curly**. All other dynamic attributes of the **Foo** class would have a reporting interval of **15**. Blank lines, lines that begin with the number sign (**#**), and lines that contain unrecognized tokens are ignored.

The reporting interval override file is read by the RMC daemon upon start. If the **refresh -s ctrmc** command is started, the file is reread. In this case, any new reporting intervals apply only to future event registrations.

Implementation specifics

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

Location

/var/ct/cfg/ctrmc.rio

ctsec.cfg File
Purpose

Provides configuration information about the authentication methods that cluster security services can use for client or 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 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 choose 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 function 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 **/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 configuration file

/var/ct/cfg/ctsec.cfg.bak
Backup of the working copy of the MAL 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

ctsec_map.global File
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. The file is 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) security mechanism. This keyword maps the HBA **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 HBA 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 HBA 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.

Example

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 is not 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 is not applied to other RSCT HBA or HBA2 identities.

unix:entilzah@*cluster*=root
The *cluster* reserved word matches any RSCT HBA or HBA2 identity containing the user name `entilzah` that originates from any host within the currently-active cluster. This grants 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 matches any RSCT HBA or HBA2 identity containing the user name `entilzah` that originates from any host within the currently-active cluster. This grants 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 matches any RSCT HBA or HBA2 identity that contains the user name `zathras` from any host to the local user identifier `zathras`. This grants 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 grants 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 matches 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 matches 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 matches 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 matches 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 grants 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.

ctsec_map.local File
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. The file is 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) security mechanism. This keyword maps the HBA **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 HBA 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 HBA 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.

Example

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 is not 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 is not applied to other RSCT HBA or HBA2 identities.

unix:entilzah@*cluster*=root
: The *cluster* reserved word matches any RSCT HBA or HBA2 identity containing the user name `entilzah` that originates from any host within the currently-active cluster. This grants 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 matches any RSCT HBA or HBA2 identity containing the user name `entilzah` that originates from any host within the currently-active cluster. This grants 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 matches any RSCT HBA or HBA2 identity that contains the user name `zathras` from any host to the local user identifier `zathras`. This grants 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 grants 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 matches 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 matches 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 matches 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 matches 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 grants 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.

netmon.cf File

Purpose

Provides network monitoring functions.

Description

The **netmon.cf** file is an optional configuration file that customers can put in place to augment the normally available ping targets with any hosts on the network that are not defined to be part of the cluster itself, and thus would not be picked up automatically, but are reachable from the cluster nodes, specifically from the IP addresses being monitored by topology services.

If the file exists in one of these paths on a node, every NIM on that node reads it when topology services are started and try to use its contents when exercising the network monitor library functions.

This file is not controlled by any cluster subsystems; it is not distributed to any other nodes in the cluster because it is added to one of them. It must be put on each node that needs it manually.

Files

/usr/sbin/rsct/samples/hats/netmon.cf
Sample version of the **netmon.cf** file

Implementation specifics

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

Location

/usr/es/sbin/cluster/netmon.cf
Location of the **netmon.cf** file in a PowerHA® environment

/var/ct/cfg/netmon.cf
Location of the **netmon.cf** file in an RSCT peer domain

unix.map File
Purpose

Defines the operating system identity used for service provider applications on the node by the UNIX host-based authentication (HBA) security mechanism.

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 default security mechanism used by the cluster security services library is the HBA mechanism. The **unix.map** file defines the identities used by the core cluster applications when interacting with the HBA mechanism. The cluster security services library expects to locate 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 softwre 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.

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

This file should not be modified unless the administrator is instructed to do so by the cluster softwre 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.

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
```

Location

/var/ct/cfg/unix.map
Contains the **unix.map** file

Files

/usr/sbin/rsct/cfg/unix.map
Default location of the **unix.map** file

Common Information Model (CIM) resource manager commands

lsassocmap Command

Purpose

Displays an association map.

Syntax

lsassocmap [**-c** *association_class*] [**-h**] [**-TV**] [*endpoint*...]

Description

The **lsassocmap** command displays the association classes available on a cluster, including the endpoints of each association. Names and endpoints of Common Information Model (CIM) association classes that are registered with the CIM resource manager are listed in table format, similar to the output of the **lscondresp** command.

If you specify the **lsassocmap** command without any parameters, it displays all of the association classes, endpoints, and roles. A *role* is the name of the class reference property in the association class definition. Roles can be used as parameters to the **-o** and **-R** flags of the **lsrsrcassoc** command to filter output. See "lsrsrcassoc Command" on page 143 for more information.

The **-c** flag limits the associations displayed to only those provided by a specific association class. You can specify any number of classes by using the *endpoint* parameter; only associations containing those classes as references (endpoints) are displayed.

Parameters

endpoint...
: Specifies one or more endpoint classes. Only association classes containing references to one of the *endpoint* classes are displayed.

Flags

-c *association_class*
: Displays associations for *association_class*.

-h
: Writes the command usage statement to standard output.

-T
: Writes the command trace messages to standard error. For your software service organization use only.

-V
: Writes the command verbose messages to standard output.

Standard output

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

Standard error

When the **-T** flag is specified, this command trace messages are written to standard error.

Exit status

0
: The command ran successfully.

1
: An error occurred with RMC.

2
: An error occurred with the command-line interface (CLI) script.

3
: An incorrect flag was specified on the command line.

4
: An incorrect parameter was specified on the command line.

5
: An error occurred with RMC that was based on incorrect command-line input.

6
: The specified association class cannot be found.

Implementation specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack and the Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
/usr/sbin/rsct/bin/lsassocmap	

Examples

To display associations that are available in a cluster, enter the following command:

```
lsassocmap
```

The following output is displayed for the AIX platform:

The following output is displayed for other platforms:

```
Association Class              Role 1          Associator 1            Role 2         Associator 2           Node

cimv2.IBMAIX_RunningOS         Antecedent      IBMAIX_OperatingSystem  Dependent      IBMAIX_ComputerSystem  c175nf14

cimv2.IBMAIX_OSProcess         GroupComponent  IBMAIX_OperatingSystem  PartComponent  IBMAIX_UnixProcess     c175nf14

cimv2.IBMAIX_CSProcessor       GroupComponent  IBMAIX_ComputerSystem   PartComponent  IBMAIX_Processor       c175nf14

cimv2.IBMAIX_HostedFileSystem  GroupComponent  IBMAIX_ComputerSystem   PartComponent  CIM_FileSystem         c175nf14


Association Class              Role 1          Associator 1            Role 2         Associator 2           Node

cimv2.Linux_RunningOS          Antecedent      Linux_OperatingSystem   Dependent      Linux_ComputerSystem   c175nf14

cimv2.Linux_OSProcess          GroupComponent  Linux_OperatingSystem   PartComponent  Linux_UnixProcess      c175nf14

cimv2.Linux_CSProcessor        GroupComponent  Linux_ComputerSystem    PartComponent  Linux_Processor        c175nf14

cimv2.Linux_HostedFileSystem   GroupComponent  Linux_ComputerSystem    PartComponent  CIM_FileSystem         c175nf14
```

lsrsrcassoc Command

Purpose

Retrieves a list of resources that are associated with a class using an association provider.

Syntax

lsrsrcassoc [**-s** "*source_selection_string*"] [**-c** *association_class*] [**-d** *association_endpoint_class*] [**-S** "*destination_selection_string*"] [**-o** *role*] [**-R** *result_role*] [**-h**] [**-TV**] *source_class_name* [*property_list*...]

Description

You can use the **lsrsrcassoc** command to learn about the relationships among CIM resources.

This command is an interface into the association query mechanism of the Common Information Model (CIM) resource manager. Association providers that are registered with the CIM resource manager are called to retrieve association data. Before using **lsrsrcassoc**, it might be helpful to run the **lsassocmap** command to find out which association classes are known to the resource monitoring and control (RMC) subsystem.

You must specify a source class name with the **lsrsrcassoc** command. With no flags specified, **lsrsrcassoc** retrieves all resources associated with every resource of this class. Flags can be used to filter which associated resources are displayed.

The command output is similar to that of **lsrsrc**. Resources associated with a source resource are displayed with their class name and one attribute per line to facilitate searching and filtering the output.

Parameters

source_class_name
: Specifies the source class in the association.

property_list
: Specifies one or more property names. Only these properties (or attributes, in RMC terminology) of associated resources are displayed. If you do not specify this parameter, all property names are displayed.

Flags

-s *source_selection_string*
: Specifies that only resources of the source class that match the selection string are used in the search for associated resources.

-S *destination_selection_string*
Specifies that only resources of the associated classes that match this selection string are displayed.

-c *association_class*
Limits the association search to only those resources tied to the source class through *association_class*.

-d *association_endpoint*
Limits the search of associated resources to just the members of this class.

-o *role* The CIM association interface defines the *role* parameter as the name of the property referring to the class on the source side of the association. Typical values for this parameter are "GroupComponent" or "PartComponent", though the specific name must come from the association class definition.

-R *result_role*
Used like the **-o** flag, except this is the name of the property that refers to the destination side of the association.

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

Standard output

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

Standard error

When the **-T** flag is specified, this command trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with the command-line interface (CLI) script.

2 An incorrect flag was specified on the command line.

3 An incorrect parameter was specified on the command line.

4 The source endpoint class was not found.

5 The destination endpoint class was not found.

6 The association class was not found.

Implementation specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack and Reliable Scalable Cluster Technology (RSCT) package for the Linux operating system.

Location

Item	Description
/usr/sbin/rsct/bin/lsrsrcassoc	

Examples

To view instances of **cimv2.IBMAIX_UnixProcess** (for AIX) and **cimv2.Linux_UnixProcess** (for Linux) that are associated with **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem** respectively on the specified node, enter:

For AIX:

```
lsrsrcassoc -c cimv2.IBMAIX_OSProcess -s 'Name=~"c175nf14"' -S \

'Name=~"emacs"' cimv2.IBMAIX_OperatingSystem Handle Parameters
```

For Linux:

```
lsrsrcassoc -c

cimv2.Linux_OSProcess -s 'Name=~"c175nf14"' -S \

'Name=~"emacs"' cimv2.Linux_OperatingSystem Handle Parameters
```

In these examples:

- **-c cimv2.IBMAIX_OSProcess** and **-c cimv2.Linux_OSProcess** are the association classes whose provider is used.
- **-s 'Name=~"c175nf14"'** is the selection string against the **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem** instances (we only want objects associated with the OS instance representing the node **c175nf14**).
- **-S 'Name=~"emacs"'** is the selection string against **cimv2.IBMAIX_UnixProcess** and **cimv2.Linux_UnixProcess** objects; only those with **Name** attributes that contain the pattern **emacs** are returned.
- **cimv2.IBMAIX_OperatingSystem** and **cimv2.Linux_OperatingSystem**, which are the "source object" parameter, are one of the classes in the association.
- **Handle Parameters** are properties that the provider is asked to return. **Handle** is the PID of the process; **Parameters** is a list of arguments to the process.

The following output is displayed:

```
Resource Persistent Attributes for cimv2.IBMAIX_UnixProcess (or cimv2.Linux_UnixProcess)

resource 1:

Handle = "2781"

Parameters = {"emacs", "-u", "foo.C"}

resource 2:

Handle = "2782"

Parameters = {"emacs", "bar.C"}

resource 3:

Handle = "2783"

Parameters = {"emacs","foo_bar.C"}

resource 4:
```

```
Handle = "2784"

Parameters = {"emacs","bar_foo.C"}

resource 5:

Handle = "2785"

Parameters = {"emacs","CIMRC.C"}

resource 6:

Handle = "26994"

Parameters = {"emacs","lsassocmap.pl"}
```

mkcimreg Command

Purpose

Registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with RMC.

Syntax

To register a class:

mkcimreg [**-I** *include_directory...*] [**-f**] [**-h**] *definition_file...*

To register a provider:

mkcimreg [**-I** *include_directory...*] [**-p** *provider_directory*] [**-h**] *registration_file...*

To compile the CIM schema:

mkcimreg [**-I** *include_directory...*] **-b** *schema_path* [**-h**]

Description

The **mkcimreg** command registers Common Information Model (CIM) classes and Common Manageability Programming Interface (CMPI) providers with the resource monitoring and control (RMC) subsystem. You can specify one or more class definition files or provider registration files with this command. Use the **-I** flag to add directories to the search path. The output from **mkcimreg** includes the names of the files that the CIM resource manager needs for working with CIM classes.

Registering classes

If you upgrade a class using the **-f** flag (that is, if the class definition has changed somehow), you must re-register all classes that are subclasses of the upgraded class so that the changes introduced into the new class propagate to its subclasses. This must be done in "descending" order, because changes propagate from parent to child. The hierarchy is:

```
------------------
Hardware_Component
------------------
        ↑
        |
------------------
Hardware_Adapter
------------------
        ↑
```

```
        |
------------------
Hardware_Ethernet
------------------
```

If, for example, **Hardware_Component** is upgraded using **mkcimreg -f**, **Hardware_Adapter** and then **Hardware_Ethernet** must both be registered afterward, in that order.

After you register any classes:

You must restart RMC.

Restarting RMC

As the final step in the CIM class registration process, the RMC subsystem must be restarted. The sequence of commands to run follows:

1. To shut down the RMC subsystem, enter:
   ```
   /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.

Flags

-I *include_directory***...**
Specifies one or more additional directories to be searched.

-f Overwrites any existing class registration data with the definitions that are provided in the class definition files.

-p *provider_directory*
Specifies a path to the provider library.

-b *schema_path*
Compiles the CIM schema file.

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

Parameters

*definition_file***...**
Specifies one or more class definition files.

*registration_file***...**
Specifies one or more provider registration files.

Security

This command requires **root** authority.

Exit Status

0 The command has run successfully.

1 An internal command error occurred.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 A class registration error occurred.

Restrictions

You cannot register a class that derives from a class that has not yet been registered.

Implementation Specifics

This command is part of the **rsct.exp.cimrm** fileset, in the **rsct.exp** package on the AIX Expansion Pack.

Standard Output

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

Standard Error

When the **-T** flag is specified, this command's trace messages are written to standard error.

Examples

1. To register the **Linux_ComputerSystem** CIM class if the class definition file is located in the **$CIMDEFS** directory, enter:

```
mkcimreg $CIMDEFS/Linux_ComputerSystem.mof
```

You must also register the CMPI provider for this class.

2. To register a CMPI provider when the registration file is located in the **$CIMDEFS** directory and the provider library is in the **$CMPIHOME** directory, enter:

   ```
   mkcimreg -p $CMPIHOME $CIMDEFS/Linux_ComputerSystemRegistration.mof
   ```

3. To compile Version 2.12 of the CIM schema, enter:

   ```
   mkcimreg -I $SCHEMA_DIR -b CIM_Schema2.12.mof
   ```

   $SCHEMA_DIR, which indicates a search path for schema MOF files, is not required, but could help **mkcimreg** find the required MOF files if they are not in the current working directory from which the command is run.

Location

/usr/sbin/rsct/bin/mkcimreg

Resource manager commands

Resource managers, along with the resource monitoring and control (RMC) subsystem, provide the administrative and monitoring capabilities of RSCT. This section explains some of the resource manager commands.

Event-response resource manager (ERRM) commands

chcondition Command

Purpose

Changes any of the attributes of a defined condition.

Syntax

To change the attributes of a condition:

chcondition [**-r** *resource_class*] [**-e** "*event_expression*"] [**-E** "*rearm_expression*"] [**-d** "*event_description*"] [**-D** "*rearm_description*"] [**-b** *interval*[**,**_max_events_][**,**_retention_period_][**,**_max_totalsize_]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,**_node_name2_...]] [**--qnotoggle** | **--qtoggle**] [**-s** "*selection_string*"] [**-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-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 to the values supplied. If the name of the condition is changed using the **-c** flag, 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 on the **chcondition** command is locked, it will not be modified; instead an error will be generated informing you that the condition is locked. To unlock a condition, you can use the **-U** flag. However, since a

condition is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition so 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.*

Flags

-b *interval*[**,***max_events*][**,***retention_period*][**,***max_totalsize*]
: Changes one or more batching-related attributes. Use commas to separate the attribute values. Do not insert any spaces between the values or the commas.

  interval specifies that the events are to be batched together for the indicated interval. Batching continues until no events are generated for an interval. Use an interval of 0 to turn batching off.

  max_events specifies that the events are to be batched together until the *max_events* number of events are generated. The interval restarts if the *max_events* number of events is reached before the interval expires.

  retention_period specifies the retention period in hours. The batched event file is saved for the time specified as the retention period. Once this time is reached, the file is automatically deleted.

  max_totalsize specifies the total size for the batched event file in megabytes (MB). The batched event file is saved until this size is reached. Once the size is reached, the file is automatically deleted.

  max_events, *retention_period*, and *max_totalsize* cannot be specified unless interval is greater than 0. When *interval* is greater than 0 and *max_events* is 0, no maximum number of events is used.

  If *retention_period* and *max_totalsize* are both specified, the batched event file is saved until the specified time or size is reached, whichever occurs first.

  If you want to change one, two, or three attribute values, you must specify a valid value or an empty field for any attributes that precede the value you want to change. You do not have to specify any values for attributes that follow the value you want to change. For example, if you only need to change the retention period, you need to specify values for *interval* and *max_events* as well. You can provide an empty field if an attribute does not need to be changed. For example, to change the retention period to 36 hours without changing the values of *interval* and *max_events*, enter:

  ```
  chcondition -b ,,36
  ```

-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.

-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.

-d "*event_description***"**
Describes the event expression.

-D "*rearm_description***"**
Describes the rearm expression.

--g 0 | 1 | 2
Specifies granularity levels that control audit logging for the condition. The levels of granularity are:

0 Enables audit logging. ERRM writes all activities to the audit log. This is the default value.

1 Enables error logging only. ERRM writes only in case of errors to the audit log.

2 Disables audit logging. ERRM does not write any records to the audit log.

-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.

-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 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.

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.

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 the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the

RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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"
```
3. To disable the recording of audit log information for the condition called "File System space used", run this command:
```
chcondition -g 2 "File System space used"
```
4. To change the maximum size of the batched event file for the condition called "File System space used" to 100 MB, run this command:
```
chcondition -b ,,,100 "File System space used"
```
5. To disable batching for the condition called "File System space used", run this command:

```
chcondition -b 0 "File System space used"
```

This command resets *max_event*, *retention_period*, and *max_totalsize*, if these values were previously specified. You must specify values for these attributes when you re-enable batching, if needed.

In the following examples, which apply to management domains, the node where 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
   ```

Location

/usr/sbin/rsct/bin/chcondition

chresponse Command

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*] [**-b** | [**-e a** | **A** | **b** | **e** | **r**]] [**-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 unlock or lock a response:

chresponse {**-U** | **-L**} [**-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 on the **chresponse** command is locked, it will not be modified; instead an error will be generated informing you that the response is locked. To unlock a response, you can use the **-U** flag. However, since a response is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a response so 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.

Flags

-a Adds the action specification to *response*.

-b Specifies that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched.

The **-b** flag cannot be specified with the **-e** flag.

-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 **displayevent**, **logevent**, **notifyevent**, and **wallevent** commands 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 | A | b | e | r
Specifies the type of event that causes the action being defined to run:

a Specifies an event. This is the default value.

A Specifies any type of event (event, error event, or rearm event).

b Specifies both an event and a rearm event.

e Specifies an error event.

r Specifies a rearm event.

More than one event type can be specified, for example: `-e ae`.

The **-e** flag cannot be specified with the **-b** flag.

-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.

-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 use only.

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

-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 preformed by this command.

-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.

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.

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 the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. 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
   ```

Location

/usr/sbin/rsct/bin/chresponse

lscondition Command
Purpose

Lists information about one or more conditions.

Syntax

lscondition [**-a**] [**-m** | **-n** | **-e**] [**-C** | **-l** | **-t** | **-d** | **-D** *delimiter*] [**-A**] [**-q**] [**-U**] [**-x**] [**-h**] [**-TV**] [*condition1* [**,***condition2*,...]**:***node_name*]

Description

The **lscondition** command lists the following information about defined conditions:

Field	Description
Name	The name of the condition.
Node	The location of the condition (for management domain scope or peer domain scope).
MonitorStatus	The status of the condition.
ResourceClass	The resource class that is monitored by this condition.
EventExpression	The expression that is used in monitoring this condition.
EventDescription	A description of the **EventExpression** field.
RearmExpression	The expression used in determining when monitoring should restart for this condition after an event has occurred.
RearmDescription	A description of the **RearmExpression** field.
SelectionString	The selection string that is applied to the attributes of **ResourceClass** to determine which resources are included in the monitoring of this condition.
Severity	The severity of the condition: critical, warning, or informational.
NodeNames	The host names of the nodes where the condition is registered.
MgtScope	The RMC scope in which the condition is monitored.
Toggle	Specifies whether the condition toggles between the event and the rearm event.
Locked	Specifies whether the resource is locked or unlocked.
EventBatchingInterval	Specifies the time in seconds that is used to determine when the accumulated events are batched together and sent to the response. A value of 0 indicates that no batching is used.
EventBatchingMaxEvents	Specifies the maximum number of events that can be in a single batch of events. A value of 0 indicates that there is no maximum if the value of **EventBatchingInterval** is not 0.
BatchedEventRetentionPeriod	Specifies the time in hours that the batched event file is kept after all associated response scripts are run.
BatchedEventMaxTotalSize	Specifies that the total saved batched event file size can't exceed a certain size in megabytes (MB) per condition. RecordAuditLog Specifies the level of detail for ERRM log entries to the audit log (ALL, Error Only, or None).

For a list of all conditions, enter the **lscondition** command without any condition names specified. A list of all the condition names is returned with the monitoring status for each condition. The default format in this case is tabular. Specifying a node name following the condition names limits the display to the conditions defined on that node. You can list all of the conditions on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope, which is determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the conditions are listed. For local scope, only conditions on the local node are listed. Otherwise, the conditions from all nodes within the domain are listed.

For all of the information about all condition names, specify the **-A** flag with the **lscondition** command. The **-A** flag causes all information about a condition to be listed when no condition names are specified.

When all the information about all conditions is listed, the default format is long. If a monitoring-status flag (**-e**, **-m**, or **-n**) is specified, the conditions with that status are listed.

When more than one condition is specified, the condition information is listed in the order in which the condition names are entered.

By default, when a condition name is specified with the **lscondition** command, all of the condition's attributes are displayed.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node.

Flags

-a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the **CT_MANAGEMENT_SCOPE** environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lscondition -a** with **CT_MANAGEMENT_SCOPE** not set will list the management domain. In this case, to list the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

-A Displays all of the attributes of the condition.

-C Displays a **mkcondition** command template based on the condition. By modifying this template, you can create new conditions. If more than one condition is specified, the template for each **mkcondition** command appears on a separate line. This flag is ignored when no conditions are specified. This flag overrides the **–l** flag.

-d Produces delimiter-formatted output. The default delimiter is a colon (**:**). Use the **–D** flag if you want to change the default delimiter.

-D *delimiter*
Produces delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

-e Lists only those conditions that are monitored in error.

-l Produces long-formatted output. Displays the condition information on separate lines.

-m Lists only those conditions that are being monitored without error.

-n Lists only those conditions that are not being monitored.

-q Does not return an error when the condition does not exist.

-t Displays the condition information in separate columns (table format).

-U Indicates whether the resource is locked.

-x Suppresses header printing.

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

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

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

Parameters

condition1 **[**,*condition2*,**...]**
Specifies the name of an existing condition that is defined on the host name *node_name*. You can

specify more than one condition name. This parameter can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring will be listed.

node_name
Specifies the node where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the **CT_MANAGEMENT_SCOPE** environment variable.

Security

The user needs read permission for the **IBM.Condition** resource class to run **lscondition**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To list all conditions and their monitoring status, run this command:

   ```
   lscondition
   ```

   The output will look like this:

   ```
    Name                       Node          MonitorStatus
   "FileSystem space used"   "nodeA"       "Monitored"
   "tmp space used"          "nodeA"       "Not monitored"
   "var space used"          "nodeA"       "Error"
   ```

2. To list general information about the condition "FileSystem space used" in long form, run this command:

   ```
   lscondition "FileSystem space used"
   ```

   The output will look like this:

   ```
   Name             = "FileSystem space used"
   Node             = "nodeA"
   MonitorStatus    = "Monitored"
   ResourceClass    = "IBM.FileSystem"
   EventExpression  = "PercentTotUsed > 99"
   EventDescription = "Generate event when space used is
                       greater than 99 percent full"
   RearmExpression  = "PercentTotUsed < 85"
   RearmDescription = "Start monitoring again after it is
                       less than 85 percent"
   SelectionString  = ""
   Severity         = "w"
   NodeNames        = "{}"
   MgtScope         = "l"
   Toggle           = "Yes"
   Locked           = "No"
   ```

3. To list the command that would create the condition "FileSystem space used", run this command:

   ```
   lscondition -C "FileSystem space used"
   ```

   The output will look like this:

   ```
   mkcondition -r IBM.FileSystem -a PercentTotUsed \
   -e "PercentTotUsed > 99" -E "PercentTotUsed < 85" \
   -d "Generate event when space used is greater than 99 percent full" \
   -D "Start monitoring after it is less than 85 percent" \
   -S w "FileSystem space used"
   ```

4. To list all conditions that have the string **space** in their names, run this command:

   ```
   lscondition space
   ```

The output will look like this:

```
Name             = "FileSystem space used"
MonitorStatus    = "Monitored"

Name             = "tmp space used"
MonitorStatus    = "Not Monitored"

Name             = "var space used"
MonitorStatus    = "Monitored"
```

5. To list the conditions that are in error, run this command:

```
lscondition -e
```

The output will look like this:

```
Name               MonitorStatus
"var space used"   "Error"
```

This example applies to clustered systems:

1. To list all conditions and their monitoring status, run this command:

```
lscondition -a
```

The output will look like this:

```
Name                      Node        MonitorStatus
"FileSystem space used"   "nodeA"     "Monitored"
"tmp space used"          "nodeB"     "Not monitored"
"var space used"          "nodeC"     "Error"
```

Location

/usr/sbin/rsct/bin/lscondition

lscondresp Command

Purpose

Lists information about a condition and any of its condition/response associations.

Syntax

To list the link between a condition and one or more responses:

lscondresp [**-a** | **-n**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-q**] [**-U**] [**-x**] [**-z**] [**-h**] [**-TV**] [*condition*[**:***node_name*] [*response1* [*response2*...]]]

To list all of the links to one or more responses:

lscondresp [**-a** | **-n**] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-q**] [**-x**] [**-z**] **-r** [**-U**] [**-h**] [**-TV**] *response1*[**:***node_name*] [*response2*...]

Description

The **lscondresp** command lists information about a condition and its linked responses. A link between a condition and a response is called a *condition/response association.* The information shows which responses are linked with a condition and whether monitoring is active for a condition and its linked response. The following information is listed:

Field	Description
Condition	The name of the condition linked with a response.
Response	The name of the response linked with the condition.
State	The state of the response for the condition. The state indicates whether a specified response is active or not.
Node	The location of the condition and the response.
Locked	Indicates whether the resource is locked or unlocked.

To list a particular condition and response, specify both the condition and the response. To list all responses to a condition, specify the condition only. To list all conditions to which a response is linked, specify the response and the **-r** flag. To list all conditions and their linked responses, do not specify any condition or response parameters.

Specifying a node name limits the display to the condition/response associations that are defined on that node. List all of the condition/response associations on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the condition/response associations are listed. For local scope, only condition/response associations on the local node are listed. For management domain scope and peer domain scope, the condition/response associations from all nodes within the domain are listed.

When neither the **-a** flag nor the **-n** flag is specified, all selected conditions for the responses are listed. Tabular format is the default.

Flags

-a Lists only those responses that are active for the condition.

-n Lists only those responses that are not active for the condition.

-l Displays the condition information and response information on separate lines (long format).

-t Displays the condition information and response information in separate columns (table format).

-d Specifies delimiter-formatted output. The default delimiter is a colon (**:**). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses *delimiter*. Use this flag to specify something other than the default colon (**:**). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

-q Does not return an error if either the *condition* or the *response* does not exist.

-U Indicates whether the resource is locked.

-x Suppresses header printing.

-z Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lscondresp -z** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-r Lists information about all of the condition/response associations for the specified responses. Use this flag to indicate that all command parameters specified are responses, not conditions.

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

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

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

Parameters

condition
The *condition* can be a condition name or a substring of a condition name. When it is a substring, any defined condition name that contains the substring and is linked to the response will be listed.

response1 **[***response2***...]**
This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring and is linked to the condition will be listed.

node_name
Specifies the node where the condition or response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs read permission for the **IBM.Association** resource class to run **lscondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in

processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

To see which resources are locked, run this command:

```
lscondresp -U
```

The output will look like this:

```
Condition             Response                 Node     State         Locked
"/tmp space used"     "E-mail root off-shift"  "nodeA"  "Not active"  "Yes"
"Page space in rate"  "E-mail root anytime"    "nodeA"  "Not active"  "No"
```

These examples apply to standalone systems:

1. To list all conditions with their linked responses, run this command:

   ```
   lscondresp
   ```

   The output will look like this:

   ```
   Condition                Response                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   "Page in Rate"           "Log event anytime"           "nodeA"    "Active"
   ```

2. To list information about the condition "FileSystem space used", run this command:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                Response                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"    "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"         "nodeA"    "Not Active"
   ```

3. To list information about the condition "FileSystem space used" for responses that are active, run this command:

   ```
   lscondresp -a  "FileSystem space used"
   ```

   The output will look like this:

```
Condition                Response                      Node       State
"FileSystem space used"  "Broadcast event on-shift"   "nodeA"    "Active"
```

4. To list information about the condition "FileSystem space used" with the linked response "Broadcast event on-shift", run this command:

   ```
   lscondresp  "FileSystem space used" "Broadcast event on-shift"
   ```

   The output will look like this:

   ```
   Condition                Response                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"   "nodeA"    "Active"
   ```

5. To list all conditions that have the string **space** in their names with their linked responses, run this command:

   ```
   lscondresp space
   ```

   The output will look like this:

   ```
   Condition                Response                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"   "nodeA"    "Active"
   "FileSystem space used   "E-mail root anytime"        "nodeA"    "Not Active"
   ```

These examples apply to management domains:

1. In this example, the condition "FileSystem space used" is defined on the management server. To list information about "FileSystem space used", run this command on the management server:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                 Response                      Node       State
   "FileSystem space used"   "Broadcast event on-shift"   "nodeB"    "Active"
   "FileSystem space used"   "E-mail root anytime"        "nodeB"    "Not Active"
   ```

2. In this example, the condition "FileSystem space used" is defined on the managed node **nodeC**. To list information about "FileSystem space used", run this command on the management server:

   ```
   lscondresp "FileSystem space used":nodeC
   ```

   The output will look like this:

   ```
   Condition                 Response                      Node     State
   "FileSystem space used"   "Broadcast event on-shift"   "nodeC"    "Active"
   "FileSystem space used"   "E-mail root anytime"        "nodeC"    "Not Active"
   ```

This example applies to a peer domain:

1. In this example, the condition "FileSystem space used" is defined in the domain. To list information about "FileSystem space used", run this command on one of the nodes in the domain:

   ```
   lscondresp "FileSystem space used"
   ```

   The output will look like this:

   ```
   Condition                Response                      Node       State
   "FileSystem space used"  "Broadcast event on-shift"   "nodeD"    "Active"
   "FileSystem space used"  "E-mail root anytime"        "nodeD"    "Not Active"
   "FileSystem space used"  "Broadcast event on-shift"   "nodeE"    "Active"
   "FileSystem space used"  "E-mail root anytime"        "nodeE"    "Not Active"
   ```

Location

/usr/sbin/rsct/bin/lscondresp

lsevent Command

Purpose

Lists event-monitoring information from the audit log.

Syntax

To list events from the audit log:

lsevent [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e a** | **r** | **b**] [**-i**] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [**-h**] [**-TV**]

To list responses from the audit log:

lsevent -r [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] [**-i**] [**-a** | **n** *node1*[**,***node2*...]] [**-h**] [**-TV**] [response [response...]]

To list events for a condition from the audit log:

lsevent [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e a** | **r** | **b**] [**-i**] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [**-h**] [**-TV**] *condition*

To list responses for a condition from the audit log:

lsevent -R [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] [-i] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [-h] [-TV] *condition* [*response* [*response*...]]

To list events and responses for a condition from the audit log:

lsevent -A [**-O** *entries*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-e** { **a** | **r** | **b** | **e** | **A** } ...] [-i] [**-a** | **n** *node1*[**,***node2*...]] [**-w** *event_node*] [-h] [-TV] condition [*response* [*response*...]]

Description

The **lsevent** command lists event-monitoring information from the audit log. The audit log contains information about monitored events or conditions, and responses that were run as a result. This information allows a system administrator to see how events are being processed. The **lsevent** command lists only the information from the audit log recorded by RSCT event response resource manager (ERRM). By using **lsevent**, you can list audit log information without knowing detailed information about ERRM audit log templates, as you would need using the **lsaudrec** command.

By default, without using options and operands, the **lsevent** command lists the events that are recorded in the audit log. These events describe the monitored events that occurred. To list the events for a particular condition, specify the condition name.

Response information can be listed separately or with the event information. Responses are run based on a condition or event occurring. Information about a response includes when it was run, what the response script was, the return code, the expected return code, standard error output, and standard output. To see standard output and the expected return code, the response resource must be defined to record it by **mkresponse** or **chresponse**. To list only response information, use the **-r** flag. You can optionally specify one or more response names to limit the number of responses listed.

To list event information and response information for a condition, you can use the **-R** and **-A** flags with a condition name. Without **-R** and **-A**, when a condition is specified, the events for the condition are listed. Specify **-R** to list the responses for the condition. You can specify one or more response names to limit the output to those responses. Specify **-A** to list the events and the responses. You can specify one or more response names to limit the response output for **-A** as well. If a condition and at least one response are specified without specifying the **-R**, **-A**, or **-r** flags, **-R** is assumed.

The type of event listed can be controlled using the **-e** flag. You can list events, rearm events, and error events for a condition. The **-w** flag can be used to list events that occurred on a particular node. The **-w**

flag has meaning when it is used in listing events. Status information is displayed when the **-i** flag is specified. When listing conditions, the status information includes showing when the condition was registered and unregistered, and when event errors occur. For response information, the status information shows that a response is about to run.

Use the **-B** and **-E** flags if you need to specify a time to limit the command output. By default, **lsevent** lists all audit log entries according to the flags specified, but you can specify a beginning time or an ending time if you are interested in a certain period. The time format is described below. The **-O** flag is used to limit the search of the audit log to the most recent records. The value used with the **-O** flag determines how many of the most recent records are searched for the other **lsevent** criteria specified. For example, using **lsevent -O 1000** causes **lsevent** to search the most recent 1000 records in the audit log for events. If **-a** or **-n** is used, **-O** cannot be used.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Parameters

condition
Specifies the name of a condition for which audit log information is listed.

response
Specifies the name of a response for which audit log information is listed.

Flags

-a Specifies that the **lsevent** command retrieves audit log information from all of the nodes in the cluster. The CT_MANAGEMENT_SCOPE environment variable determines the scope of the cluster. If CT_MANAGEMENT_SCOPE is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and CT_MANAGEMENT_SCOPE is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set CT_MANAGEMENT_SCOPE to **2**.

-A Specifies that event and response information for a condition is to be listed.

-B *MMddhhmmyyyy*
Specifies to list the audit log entries beginning at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-e a | r | b | e | A
Specifies the type of event to list from the audit log:

a Lists events from conditions. It is the default setting.

r Lists rearm events from conditions.

b List events and rearm events from conditions.

e Lists response information that is triggered by error events. This setting is meaningful only when **-r**, **-R**, or **-A** is specified.

A Lists all types of events (events, rearm events, and error events).

More than one event type can be specified, for example: **-e ae**.

-E *MMddhhmmyyyy*
Specifies to list the audit log entries up to or ending at the time indicated. This time indicates when the audit log entry was created. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified.

-i Specifies that status information for a condition or response is to be listed. The status information includes information about event registration, event errors, and responses about to be run.

n *node1***[,***node2***...]**
Specifies the node or nodes from which the audit log information is to be retrieved. If node is not specified, the local node is used. *node* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

-O *entries*
Specifies that only the latest entries in the audit log are searched for information. *entries* determines how many of the most recent records are search for the other **lsevent** criteria specified. For example, using **-O 1000** causes the **lsevent** command to search the most recent 1000 records in the audit log for events.

-r Specifies that all command parameters are response names and that response information is to be returned for the responses specified. There are no condition names in the parameter list. If no response names are specified, then information is listed for all responses.

-R Specifies that only the response information for a condition is to be listed.

-w *event_node*
Specifies the node on which the event occurred. This flag is only meaningful in listing events.

-h Writes this command usage statement to standard output.

-T Writes the command trace messages to standard error. For your software service organization use only.

-V Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Standard output

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

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Restrictions

If you are using the **lsevent** command, you must have read access to the ERRM audit log resource on each node from which records are to be listed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Implementation specifics

This command is part of the **rsct.core** fileset for the AIX operating system and **rsct.core-3.1.0.0-0.***platform***.rpm** package for the Linux, Solaris, and Windows platforms, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Location

/usr/sbin/rsct/bin/lsevent

Examples

1. To list the information for events that occurred, enter:
   ```
   lsevent
   ```
2. To list the event information for a condition named **Condition1**, enter:
   ```
   lsevent Condition1
   ```
3. To list the event response information, enter:
   ```
   lsevent -r
   ```
4. To list the event response information for a response named **Response1**, enter:
   ```
   lsevent -r Response1
   ```
5. To view the output of the event response named **Response1**, which is defined to save its output, enter:

```
lsevent -r Response1
```

6. To see the events found in the latest 1000 audit log records, enter:
```
lsevent -O 1000
```
7. To list the rearm event information for a condition named **Condition1**, enter:
```
lsevent -e r Condition1
```

lsresponse Command
Purpose

Lists information about one or more responses.

Syntax

lsresponse [**-a**] [**-C** | **-l** | **-t** | **-d** | **-D** *delimiter*] [**-A**] [**-q**] [**-U**] [**-x**] [**-b**] [**-h**] [**-TV**] [*response1*[**,***response2*,...] **:***node_name*]

Description

The **lsresponse** command lists the following information about defined responses:

Field	Description
ResponseName	The name of the response.
Node	The location of the response.
Action	The name of an action.
DaysOfWeek	The days of the week when the action can be run. **DaysOfWeek** and **TimeOfDay** together define the interval when the action can be run. The values for the days can be separated by plus signs (**+**) or displayed as a range of days separated by a hyphen (**-**). Multiple **DaysOfWeek** values are separated by commas (**,**). The number of **DaysOfWeek** values must match the number of **TimeOfDay** values. The values for each day follow: **1** Sunday **2** Monday **3** Tuesday **4** Wednesday **5** Thursday **6** Friday **7** Saturday
TimeOfDay	The time range when **Action** can be run, consisting of the start time followed by the end time separated by a hyphen. **DaysOfWeek** and **TimeOfDay** together define the interval when the action can be run. The time is in 24-hour format (HHMM), where the first two digits represent the hour and the last two digits represent the minutes. Multiple **TimeOfDay** values are separated by commas (**,**). The number of **DaysOfWeek** values must match the number of **TimeOfDay** values.
ActionScript	The script or command to run for the action.
ReturnCode	The expected return code for **ActionScript**.
CheckReturnCode	Indicates whether the actual return code for **ActionScript** is compared to its expected return code. The values are: **y** (yes) and **n** (no).
EventType	The type of event that causes the action to be run: event, rearm event, or both.
StandardOut	Indicates whether standard output is directed to the audit log. The values are: **y** (yes) and **n** (no).
EnvironmentVars	Indicates any environment variables that will be set before the action is run.
UndefRes	Indicates whether the action is to be run if a monitored resource becomes undefined. The values are: **y** (yes) and **n** (no).
Locked	Indicates whether the resource is locked or unlocked.
EventBatching	Indicates whether the response action supports event batching.

To get a list of all response names, run the **lsresponse** command alone without any response names specified. A list of all response names is returned. The default format in this case is tabular.

Specifying a node name after the response names limits the display to the responses defined on that node. List all of the responses on a node by specifying a colon (**:**) followed by the node name. The node name is a node within the management scope determined by the CT_MANAGEMENT_SCOPE environment variable. The management scope determines the list of nodes from which the responses are listed. For local scope, only responses on the local node are listed. Otherwise, the responses from all nodes within the domain are listed.

To see all the information about all response names, specify the **-A** flag with the **lsresponse** command. The **-A** flag causes all information about a response to be listed when no response names are specified. When all of the information about all responses is listed, the long format is the default.

When more than one response is specified, the response information is listed in the order in which the responses are entered.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

Flags

-a Specifies that this command applies to all nodes in the cluster. The cluster scope is determined by the CT_MANAGEMENT_SCOPE environment variable. If it is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management and peer domain exist, **lsresponse -a** with CT_MANAGEMENT_SCOPE not set will list the management domain. In this case, to list the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A Displays all of the attributes of the response.

-b Displays only the responses that support event batching.

-C Displays the **mkresponse** command that can be used to create the response and one of its actions. If more than one response is specified, each **mkresponse** command appears on a separate line. This flag is ignored when no responses are specified. This flag overrides the **-l** flag.

-d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **–D** flag if you wish to change the default delimiter.

-D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default, colon (:). For example, when the data to be displayed contains colons, use this flag to specify another delimiter of one or more characters.

-l Displays the response information on separate lines (long form).

-q Does not return an error when **response** does not exist.

-t Displays the response information in separate columns (table form).

-U Indicates whether the resource is locked.

-x Suppresses headers when printing.

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

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

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

Parameters

response1[,*response2*,...]
: This parameter can be a response name or a substring of a response name. You can specify more than one response name. When it is a substring, any defined response name that contains the substring is listed.

node_name
: Specifies the node where the response is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

Security

The user needs read permission for the **IBM.EventResponse** resource class to run **lsresponse**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
: Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
: When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
: Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

  0 Specifies *local* scope.

  1 Specifies *local* scope.

  2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To list all of the responses, run this command:

   ```
   lsresponse
   ```

   The output will look like this:

   ```
   ResponseName
   "E-mail root anytime"
   "E-mail root first shift"
   "Critical notifications"
   "Generate SNMP trap"
   ```

2. To see which resources are locked, run this command:

   ```
   lsresponse -U
   ```

   The output will look like this:

   ```
   ResponseName                    Node      Locked
   "Broadcast event on-shift"      "nodeA"   "No"
   "E-mail root off-shift"         "nodeA"   "No"
   "E-mail root anytime"           "nodeA"   "No"
   "Log event anytime"             "nodeA"   "No"
   "Informational notifications"   "nodeA"   "No"
   "Warning notifications"         "nodeA"   "No"
   "Critical notifications"        "nodeA"   "No"
   "Generate SNMP trap"            "nodeA"   "No"
   ```

3. To list general information about the response "Critical notifications", run this command:

   ```
   lsresponse "Critical notifications"
   ```

   The output will look like this:

   ```
   ResponseName     = "Critical notifications"
   Node             = "nodeA"
   Action           = "Log Critical Event"
   DaysOfWeek       = 1+2+7
   TimeOfDay        = 0000-2400
   ActionScript     = "/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"
   ```

Location

/usr/sbin/rsct/bin/lsresponse

mkcondition Command
Purpose

Creates a new condition definition which can be monitored.

Syntax

mkcondition -r *resource_class* **-e** "*event_expression*" [**-E** "*rearm_expression*"] [**-d** "*event_description*"] [**-D** "*rearm_description*"] [**-b** *interval*[**,***max_events*][**,***retention_period*][**,***max_totalsize*]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,***node_name2*...]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s** "*selection_string*"] [**-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-h**] [**-TV**] *condition*

mkcondition -c *existing_condition*[**:***node_name*] [**-r** *resource_class*] [**-e** "*event_expression*"] [**-E** "*rearm_expression*"] [**-d** "*event_description*"] [**-D** "*rearm_description*"] [**-b** *interval*[**,***max_events*][**,***retention_period*][**,***max_totalsize*]] [**-m l** | **m** | **p**] [**-n** *node_name1*[**,***node_name2*...]] [**-p** *node_name*] [**--qnotoggle** | *--qtoggle*] [**-s** "*selection_string*"] [**-S c** | **w** | **i**] [**-g 0** | **1** | **2**] [**-h**] [**-TV**] *condition*

Description

The **mkcondition** command creates a new condition with the name specified by the condition parameter. The condition is used to monitor a resource for the occurrence of the condition (or event). Use the **mkresponse** command to define one or more responses to an event. You can then link the conditions to the responses using the **mkcondresp** command, or you can use the **startcondresp** command to link the responses and start monitoring.

Using the **-b** flag, multiple events can be batched or grouped together and passed to a response. The grouping of events is by the time span in which they occur. In addition, the grouping can be done such that a specified maximum number of events are grouped within the time span. A response that handles batched events must be defined as supporting batched events.

In a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the condition definition. If you are using **mkcondition** on the management server and you want the condition to be defined on the management server, do *not* specify the **-p** flag. If the **-p** flag is not specified, the condition is defined on the local node. If the node where the condition will be defined is:

- in a cluster of nodes, the condition can monitor resources on more than one node. Use the **-n** flag to specify the nodes on which the condition will be monitored.
- the management server in a management domain, a management scope (**-m**) of local (**l**) or management domain (**m**) can be specified to indicate how the condition applies. The selection string will be evaluated using the entire management domain when management scope is set to the management domain and the node is the management server.
- a managed node in a management domain, only a management scope (**-m**) of local (**l**) can be used.
- in a peer domain, a management scope (**-m**) of peer domain (**p**) or local (**l**) can be used to indicate how the condition and the selection string apply.
- in both a management domain and a peer domain, a management scope (**-m**) of management domain (**m**), peer domain (**p**), or local (**l**) can be used to indicate how the condition and its selection string apply.

To lock a condition so it cannot be modified or removed, use the **chcondition** command (with its **-L** flag).

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference.*

Flags

-b *interval*[**,***max_events*][**,***retention_period*][**,***max_totalsize*]

Specifies one or more batching-related attributes. Use commas to separate the attribute values. Do not insert any spaces between the values or the commas.

interval specifies that the events are to be batched together for the indicated interval. Batching continues until no events are generated for an interval. Use an interval of 0 to turn batching off.

max_events specifies that the events are to be batched together until the *max_events* number of events are generated. The interval restarts if the *max_events* number of events is reached before the interval expires.

retention_period specifies the retention period in hours. The batched event file is saved for the time specified as the retention period. Once this time is reached, the file is automatically deleted.

max_totalsize specifies the total size for the batched event file in megabytes (MB). The batched event file is saved until this size is reached, Once the size is reached, the file is automatically deleted.

max_events, *retention_period*, and *max_totalsize* cannot be specified unless interval is greater than 0.

When *interval* is greater than 0 and *max_events* is 0, no maximum number of events is used.

If *retention_period* and *max_totalsize* are both specified, the batched event file is saved until the specified time or size is reached, whichever occurs first.

If you want to change one, two, or three attribute values, you must specify a valid value or an empty field for any attributes that precede the value you want to change. You do not have to specify any values for attributes that follow the value you want to change. For example, if you

only need to change the retention period, you need to specify values for interval and *max_events* as well. You can provide an empty field if an attribute does not need to be changed. To change the retention period to 36 hours without changing the values of interval and *max_events*, enter:

```
mkcondition -c existing_condition -b ,,36
```

-c *existing_condition*[**:***node_name*]
Copies an existing condition. The existing condition is defined on *node_name*. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable. If any other flags are specified, update the new condition as indicated by the flags. Links with responses are not copied.

-d "*event_description*"
Describes the event expression.

-D "*rearm_description*"
Describes the rearm expression.

-e "*event_expression*"
Specifies an *event expression*, which determines when an event occurs. An event expression consists of a dynamic attribute or a persistent attribute of *resource_class*, a mathematical comparison symbol (or **<**, for example), and a constant. When this expression evaluates to TRUE, an event is generated.

-E "*rearm_expression*"
Specifies a rearm expression. After *event_expression* has evaluated to True and an event is generated, the rearm expression determines when monitoring for the event expression will begin again. Typically, the rearm expression prevents multiple events from being generated for the same event evaluation. The rearm expression consists of dynamic attributes or persistent attributes of *resource_class*, mathematical comparison symbols (> or <, for example), logical operators (|| or &&), constants, and an optional qualifier.

--g 0 | 1 | 2
Specifies granularity levels that control audit logging for the condition. The levels of granularity are:

0 Enables audit logging. ERRM writes all activities to the audit log. This is the default.

1 Enables error logging only. ERRM writes only in case of errors to the audit log.

2 Disables audit logging. ERRM does not write any records to the audit log.

-m l | m | p
Specifies the management scope to which the condition applies. The management scope determines how the condition is registered and how the selection string is evaluated. The scope can be different from the current configuration, but monitoring cannot be started until an appropriate scope is selected. The valid values are:

l Specifies *local* scope. This is the default. The condition applies only to the local node (the node where the condition is defined; see the **-p** flag). Only the local node is used in evaluating the selection string.

m Specifies *management domain* scope. The condition applies to the management domain in which the node where the condition is defined belongs (see the **-p** flag). All nodes in the management domain are used in evaluating the selection string. The node where the condition is defined must be the management server in order to use management domain scope.

p Specifies *peer domain* scope. The condition applies to the peer domain in which the node where the condition is defined belongs (see the **-p** flag). All nodes in the peer domain are used in evaluating the selection string.

-n *node_name1*[**,***node_name2***...**]
Specifies the host name for a node (or a list of host names separated by commas for multiple nodes) where this condition will be monitored. Node group names can also be specified, which are expanded into a list of node names.

You must specify the **-m** flag with a value of **m** or **p** if you want to use the **-n** flag. This way, you can monitor conditions on specific nodes instead of the entire domain.

The host name does not have to be online in the current configuration, but once the condition is monitored, the condition will be in error if the node does not exist. The condition will remain in error until the node is valid.

-p *node_name*
Specifies the name of the node where the condition is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

If you are using **mkcondition** on the management server and you want the condition to be defined on the management server, do *not* specify the **-p** flag.

--qnotoggle
Specifies that monitoring does not toggle between the event expression and the rearm expression, but instead the event expression is always evaluated.

--qtoggle
Specifies that monitoring toggles between the event expression and the rearm expression.

-r *resource_class*
Specifies the resource class to be monitored by this condition. You can display the resource class names using the **lsrsrcdef** command.

-s "*selection_string***"**
Specifies a selection string that is applied to all of the *resource_class* attributes to determine which resources should be monitored by the *event_expression*. The default is to monitor all resources within the *resource_class*. The resources used to evaluate the selection string is determined by the management scope (the **-m** flag). The selection string must be enclosed within double or single quotation marks. For information on how to specify selection strings, see the *RSCT: Administration Guide* .

-S c | w | i
Specifies the severity of the event:
c Critical
w Warning
i Informational (the default)

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

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

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

Parameters

condition
The *condition* name is a character string that identifies the condition. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the **IBM.Condition** resource class to run **mkcondition**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90% and to rearm when the percentage is back down below 85%, enter:

   ```
   mkcondition -r IBM.FileSystem  \
   -e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
   "FileSystem space used"
   ```

2. To define a condition with the name "tmp space used" to check for percentage of space used greater than 90% for **/tmp** and to rearm when the percentage is back down below 85%, including comments, enter:

   ```
   mkcondition -r IBM.FileSystem  \
   -e "PercentTotUsed > 90" -E "PercentTotUsed < 85" \
   -d "Generate event when tmp > 90% full"  \
   -D "Restart monitoring tmp again after back down < 85% full"\
   -s 'Name=="/tmp"'  "tmp space used"
   ```

3. To define a condition with the name "Space used" as a copy of "FileSystem space used", enter:

   ```
   mkcondition -c "FileSystem space used"  "Space used"
   ```

4. To define a condition with the name "var space used" as a copy of "tmp space used", but change the selection to **/var**, enter:

   ```
   mkcondition -c "tmp space used" -s 'Name=="/var"' \
   "var space used"
   ```

5. To define a condition with the name "vmstat is running" to monitor when user **joe** is running the **vmstat** program in a 64-bit environment, enter:

   ```
   mkcondition -r "IBM.Program" \
   -e "Processes.CurPidCount > 0" -E "Processes.CurPidCount  <= 0"  \
   -d "Generate event when user starts vmstat" \
   -D "Restart monitoring when vmstat is terminated" \
   -s ProgramName == \"vmstat64\" && Filter==\"ruser==\\\"joe\\\"\"" \
   -S "i" -m "l" "vmstat is running"
   ```

6. To define a condition with the name "myscript terminated" to monitor when a script has ended, enter:

   ```
   mkcondition -r "IBM.Program" \
   -e "Processes.CurPidCount <= 0" -E "Processes.CurPidCount  > 0"  \
   -d "Generate event when myscript is down" \
   -D "Rearm the event when myscript is running" \
   -s ProgramName == \"ksh\" && Filter == 'args[1]==\"/home/joe/myscript\"'" \
   -m "l" "myscript terminated"
   ```

   In this example, **args** represents the array of argument strings that was passed to **main**. Because this is an array, **args[1]** references the first argument after the program name. Use the **ps -el** command to determine the **ProgramName**. See the **lsrsrcdef** command for more information.

7. To batch together a maximum of 20 events at a time that come from a sensor named **DBInit** in 60-second intervals, enter:

   ```
   mkcondition -r "IBM.Sensor" \
   -e "Int32 < 0" -E "Int32 > 0"  -b 60,20 \
   -s "Name == \"DBInit\""  "DBInit Sensor"
   ```

8. To define a condition with the name tmp space used to check for percentage of space used greater than 90% for **/tmp** for at least seven out of the last 10 observations, including comments, enter:

```
mkcondition -r IBM.FileSystem \
-e "PercentTotUsed > 90 __QUAL_COUNT(7,10)" \
-d "Generate event when tmp > 90% full for 7 out of 10 last
\observations" \ -s 'Name=="/tmp"' "tmp space used"
```

9. To define a condition with the name adapter stability to check for adapter status that has changed four times within one minute, including comments, enter:

```
mkcondition -r IBM.NetworkInterface \
-e "OpState != OpState@P __QUAL_RATE(4,60)" \
-d "Generate event when OpState is changed 4 times within 1 minute" \
"adapter stability"
```

10. To define a condition for a batched event called tmp space used to check the percentage of space used by **/tmp** that is greater then 90%, with a batch interval of 5 and a batch event file retention period of 72 hours, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -b 5,,72 "tmp space used"
```

11. To define a condition called tmp space used to check that percentage of space used by **/tmp** that is greater then 90%, with audit logging enabled only in case of errors, enter:

```
mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" -g 1 "tmp space used"
```

These examples apply to management domains:

1. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command on the management server:

```
 mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m d "FileSystem space used"
```

2. To define a condition with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes **nodeA** and **nodeB** in the domain, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m p \
"FileSystem space used"
```

3. To define a condition with the name "nodeB FileSystem space used" on **nodeB** to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the condition with local scope, run this command on the management server:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l -p nodeB \
"nodeB FileSystem space used"
```

4. To define a condition with the name "local FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node, run this command on a managed node:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l "local FileSystem space used"
```

These examples apply to peer domains:

1. To define a condition on **nodeA** with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor all nodes in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m p -p nodeA "FileSystem space used"
```

2. To define a condition on **nodeC** with the name "FileSystem space used" to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor nodes **nodeA** and **nodeB** in the domain, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -n nodeA,nodeB -m p -p nodeC \
"FileSystem space used"
```

3. To define a condition with the name "local FileSystem space used" on **nodeB** to check for percentage of space used greater than 90%, to rearm when the percentage is back down below 85%, and to monitor the local node only, run this command:

```
mkcondition -r IBM.FileSystem  -e "PercentTotUsed > 90" \
-E "PercentTotUsed < 85" -m l -p nodeB "local FileSystem space used"
```

Location

/usr/sbin/rsct/bin/mkcondition

mkcondresp Command

Purpose

Creates a link between a condition and one or more responses.

Syntax

mkcondresp [**-h**] [**-TV**] *condition*[**:***node_name*] *response1* [*response2*...]

Description

The **mkcondresp** command creates a link between a condition and one or more responses. A link between a condition and a response is called a *condition/response association.* This command creates one or more condition/response associations; it does not start monitoring. In a cluster environment, the condition and the response must be defined on the same node. You can start monitoring for this condition and its linked responses later using the **startcondresp** command.

To lock a condition/response association, use the **-L** flag of the **rmcondresp**, **startcondresp**, or **stopcondresp** command.

Flags

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

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

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

Parameters

condition
Specifies the name of the condition to be linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response1 **[***response2***...]**
Specifies one or more response names. All responses are linked to *condition*.

Security

The user needs write permission for the **IBM.Association** resource class to run **mkcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To link the condition "FileSystem space used" to the response "Broadcast event on-shift", run this command:
   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift"
   ```
2. To link the condition "FileSystem space used" to the responses "Broadcast event on-shift" and "E-mail root anytime", run this command:
   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

These examples apply to management domains:

1. To link the condition "FileSystem space used" on the management server to the response "Broadcast event on-shift" (also on the management server), run this command on the management server:
   ```
   mkcondresp "FileSystem space used" "Broadcast event on-shift"
   ```
2. To link the condition "FileSystem space used" on the management server to the response "Broadcastevent on-shift", run this command on one of the nodes in the domain:
   ```
   mkcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To link the condition "FileSystem space used" on node **nodeA** to the response "Broadcastevent on-shift" (also on **nodeA**), run this command on one of the nodes in the domain:
   ```
   mkcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/mkcondresp

mkresponse Command
Purpose

Creates a new response definition.

Syntax

To create a response with no actions:

mkresponse [**-b**] [**-p** *node_name*] [**-h**] [**-TV**] *response*

To create a response with one action:

mkresponse -n *action* [**-d** *days_of_week*[**,***days_of_week*...]] [**-t** *time_of_day*[**,***time_of_day*...]] **-s** *action_script* [**-r** *return_code*] [**-b** | [**-e a** | **A** | **b** | **e** | **r**]] [**-o**] [**-E** *env_var*=*value*[**,***env_var*=*value*...]] [**-u**] [**-p** *node_name*] [**-h**] [**-TV**] *response*

To copy a response:

mkresponse -c *existing_response*[**:***node_name*] [**-p** *node_name*] [**-h**] [**-TV**] *response*

Description

The **mkresponse** command creates a new response definition with the name specified by the *response* parameter. One action can also be specified when the response is defined. Actions define commands to be run when the response is used with a condition and the condition occurs. The action defines days of the week when the action can be used, the time of day for those days of the week, the script or command to be run, what type of event causes the command to be run, the expected return code of the script or command, and whether to keep standard output. The days and times are paired so that different times can be specified for different days. A response with no actions only logs the events.

Use the **-b** flag to specify that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched. The **-b** flag cannot be specified with the **-e** flag.

In a cluster environment, use the **-p** flag to specify the node in the domain that is to contain the response definition. If you are using **mkresponse** on the management server and you want the response to be defined on the management server, do *not* specify the **-p** flag. If the **-p** flag is not specified, the response is defined on the local node.

Use the **chresponse** command to add actions to a response or to remove actions from a response. Use the **startcondresp** command to start monitoring. The **startcondresp** command links a response to a condition, if they are not already linked.

To lock a response so it cannot be modified or removed, use the **chresponse** command with the **-L** flag.

Flags

-b Specifies that the response, and all actions to be defined in this response, support event batching. For event batching, multiple events can be batched or grouped together and passed to a response. The actions of the response are directed to a file that contains the details for the batched events. A response that supports event batching can only be used for conditions that specify the events are to be batched.

An event response can be created for batched event conditions without an action script.

The **-b** flag cannot be specified with the **-e** flag.

-c *existing_response*[**:***node_name*]
Copies an existing response. Links with conditions are not copied. The existing response is defined on the node known as *node_name* in a cluster. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable. If any other flags are specified, update the new response as indicated by the flags.

-d *days_of_week*

Specifies the days of the week when the action being defined can be run. *days_of_week* and *time_of_day* together define the interval when the action can be run.

Enter the numbers of the days separated by a plus sign (+) or as a range of days separated by a hyphen (-). More than one *days_of_week* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default is all days. If no value is specified but a comma is entered, the default value is used. The values for each day follow:

1 Sunday
2 Monday
3 Tuesday
4 Wednesday
5 Thursday
6 Friday
7 Saturday

-e a | A | b | e | r
Specifies the type of event that causes the action being defined to run:

a Specifies an event. This is the default.

A Specifies any type of event (event, error event, or rearm event).

b Specifies an event and a rearm event.

e Specifies an error event.

r Specifies a rearm event.

More than one event type can be specified, for example: **-e ae**. The **-e** flag cannot be specified with the **-b** flag.

-E *env_var*=*value*[**,***env_var*=*value***...**]
Specifies any environment variables to be set before running the action. If multiple *env_var*=*value* variables are specified, they must be separated by commas.

-n *action*
Specifies the name of the action being defined. Only one action can be defined when the response is created. Use the **chresponse** command to add more actions to the response.

-o Directs all standard output from *action_script* to the audit log. The default is not to keep standard output. Standard error is always directed to the audit log.

-p *node_name*
Specifies the name of the node where the response is defined. This is used in a cluster environment and the node name is the name by which the node is known in the domain. The default *node_name* is the local node on which the command runs. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

If you are using **mkresponse** on the management server and you want the response to be defined on the management server, do *not* specify the **-p** flag.

-r *return_code*
Specifies the expected return code for *action_script*. If the expected return code is specified, the actual return code of *action_script* is compared to the expected return code. A message is written to the audit log indicating whether they match. If the **-r** flag is not specified, the actual return code is written to the audit log, and no comparison is performed.

-s *action_script*
Specifies the fully-qualified path for the script or command to run for the action being defined. See the **logevent**, **notifyevent**, and **wallevent** commands for descriptions of the predefined response scripts provided with the application.

-t *time_of_day*
Specifies the time range when *action* can be run, consisting of the start time followed by the end time, separated by a hyphen. *days_of_week* and *time_of_day* together define the interval when the action can be run.

The time is in 24-hour format (HHMM) where the first two digits represent the hour and the last two digits represent the minutes. The start time must be less than the end time because the time is specified by day of the week. More than one *time_of_day* parameter can be specified, but the parameters must be separated by a comma (,). The number of *days_of_week* parameters specified must match the number of *time_of_day* parameters specified. The default value is 0000-2400. If no value is specified but a comma is entered, the default value is used.

-u Specifies that the action is to be run when a monitored resource becomes undefined.

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

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

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

Parameters

response

The *response* name is a character string that identifies the response. If the name contains spaces, it must be enclosed in quotation marks. A name cannot consist of all spaces, be null, or contain embedded double quotation marks.

Security

The user needs write permission for the **IBM.EventResponse** resource class to run **mkresponse**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT

Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT

When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To define a response with the name "Log event in audit log", run this command:

   ```
   mkresponse "Log event in audit log"
   ```

2. To define a response with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/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"
   ```

7. To define a batching-capable response called "Batched Event Response" without an action script, enter:

   ```
   mkresponse -b "Batched Event Response"
   ```

These examples apply to management domains:

1. To define a response on the management server with the name "E-mail root anytime" that has an action named "E-mail root", to be used any time Saturday and Sunday and uses the command **/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"
```

Location

/usr/sbin/rsct/bin/mkresponse

rmcondition Command

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.

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.

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.

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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP

address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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"
```

Location

/usr/sbin/rsct/bin/rmcondition

rmcondresp Command
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 {**-U** | **-L**} [**-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.

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.

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

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

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

-U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

-L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

Parameters

condition

Specifies the name of the condition linked to the response. The condition is always specified first 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.

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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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
   ```

Location

/usr/sbin/rsct/bin/rmcondresp

rmresponse Command
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.

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.

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.

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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To 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
```

Location

/usr/sbin/rsct/bin/rmresponse

startcondresp Command
Purpose

Starts monitoring a condition that has one or more linked responses.

Syntax

To start monitoring a condition:

startcondresp [**-h**] [**-TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

startcondresp {**-U** | **-L**} [**-h**] [**-TV**] *condition*[**:***node_name*] *response*

Description

The **startcondresp** command starts the monitoring of a condition that has a linked response. A link between a condition and a response is called a *condition/response association.* In a cluster environment, the condition and the response must be defined on the same node. After monitoring is started, when the condition occurs, the response is run. If no responses are specified, monitoring is started for all responses linked to the condition. This causes all of the linked responses to run when the condition occurs. If more than one response is specified, monitoring is started only for those linked responses.

If one or more responses are specified and the responses are not linked with the condition, the **startcondresp** command links the specified responses to the condition, and monitoring is started. Use the **mkcondresp** command to link a response to a condition without starting monitoring.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be started by the **startcondresp** command. If the condition/response association you specify on the **startcondresp** command is locked, it will not be started; instead an error will be generated informing you that this condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. However, because a condition/response association is typically locked because it is essential for system software to work properly, you should exercise caution before unlocking it. To lock a condition/response association so it cannot be started, stopped, or removed, reissue this command using its **-L** flag.

Flags

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

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

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

-U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system

software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

-L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

Parameters

condition
Specifies the name of the condition linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response
Specifies the name of one or more responses. Specifying more than one response links the responses to the condition if they are not already linked and starts monitoring for the specified responses.

Security

The user needs write permission for the **IBM.Association** resource class to run **startcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in

processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To start monitoring for the condition "FileSystem space used " by using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command:
   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```
2. To start monitoring for the condition "FileSystem space used " by using all of its linked responses, run this command:
   ```
   startcondresp "FileSystem space used"
   ```
3. To start monitoring for the condition "FileSystem space used " by using the response "Broadcast event on-shift" and "E-mail root anytime", whether or not they are linked with the condition, run this command:
   ```
   startcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
   ```

These examples apply to management domains:

1. To start monitoring for the condition "FileSystem space used" on the management server using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command on the management server:
   ```
   startcondresp  "FileSystem space used" "Broadcast event on-shift"
   ```
2. To start monitoring for the condition "FileSystem space used" on the managed node **nodeB** using the response "Broadcast event on-shift", whether or not the response is linked with the condition, run this command on the management server:
   ```
   startcondresp  "FileSystem space used":nodeB "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To start monitoring for the condition "FileSystem space used" on **nodeA** in the domain using the response "Broadcast event on-shift" (also on **nodeA** in the domain), whether or not the response is linked with the condition, run this command on any node in the domain:
   ```
   startcondresp  "FileSystem space used":nodeA "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/startcondresp

stopcondresp Command
Purpose

Stops the monitoring of a condition that has one or more linked responses.

Syntax

To stop monitoring a condition:

stopcondresp [**-q**] [**-h**] [**-TV**] *condition*[**:***node_name*] [*response* [*response*...]]

To unlock or lock the condition/response association:

stopcondresp {**-U** | **-L**} [**-h**] [**-TV**] *condition*[**:***node_name*] *response*

Description

The **stopcondresp** command stops the monitoring of a condition that has one or more linked responses. If no response is specified, all of the linked responses for the condition are stopped. If one or more responses is specified, only those responses that are linked to the condition are stopped. When the condition occurs, the response is not run. If no responses are active for a condition, the condition is no longer monitored.

If a particular condition/response association is needed for system software to work properly, it may be locked. A locked condition/response association cannot be stopped by the **stopcondresp** command. If the condition/response link you specify on the **stopcondresp** command is locked, it will not be stopped; instead an error will be generated informing you that the condition/response association is locked. To unlock a condition/response association, you can use the **-U** flag. A condition/response association is typically locked because it is essential for system software to work properly, so you should exercise caution before unlocking it.

Flags

-q Does not return an error when either *condition* or *response* does not exist or when the *condition* linked with *response* is not being monitored.

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

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

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

-U Unlocks a condition/response association so it can be started, stopped, or removed. If a condition/response association is locked, this is typically because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking it. When unlocking a condition/response association using the **-U** flag, no other operation can be performed by this command.

-L Locks a condition/response association so it cannot be started, stopped, or removed. When locking a condition/response association using the **-L** flag, no other operation can be performed by this command.

Parameters

condition
Specifies the name of the condition linked to the response. The condition is always specified first.

node_name
Specifies the node in the domain where the condition is defined. If *node_name* is not specified, the local node is used. *node_name* is a node within the scope determined by the CT_MANAGEMENT_SCOPE environment variable.

response
Specifies the names of one or more responses. Monitoring is stopped for the specified responses. (If a specified response is not linked to the condition, it is ignored.)

Security

The user needs write permission for the **IBM.Association** resource class to run **stopcondresp**. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the event-response resource manager (ERRM). The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

These examples apply to standalone systems:

1. To stop monitoring for the condition "FileSystem space used " which has the response "Broadcast event on-shift" linked with it, run this command:
   ```
   stopcondresp "FileSystem space used" "Broadcast event on-shift"
   ```
2. To stop monitoring for the condition "FileSystem space used " using all of its linked responses, run this command:
   ```
   stopcondresp "FileSystem space used"
   ```

This example applies to management domains:

1. To stop monitoring for the condition "FileSystem space used " on the managed node **nodeB** which has the response "Broadcast event on-shift" linked with it, run this command on the management server:
   ```
   stopcondresp "FileSystem space used:nodeB" "Broadcast event on-shift"
   ```

This example applies to peer domains:

1. To stop monitoring for the condition "FileSystem space used " on the node **nodeA** which has the response "Broadcast event on-shift" linked with it, run this command on any node in the domain:
   ```
   stopcondresp "FileSystem space used:nodeA" "Broadcast event on-shift"
   ```

Location

/usr/sbin/rsct/bin/stopcondresp

ERRM scripts

displayevent

Purpose

Sends a message about an event or a rearm event to a specified X-window display.

Syntax

displayevent [**-h**] *x-display*

Description

The **displayevent** script sends a message about an event or a rearm event to a specified X-window display. 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.

The **displayevent** script can be used as an action that an event-response resource runs. You can also use this 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*.

This script returns event information about the following ERRM environment variables, which are described in *RSCT: Administration Guide*:

```
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
  Time:  local time
```

This example is a template of the message that appears on the specified X-window display when an event or a rearm event occurs for which **displayevent** is a response action. For the **Time:** field, this script returns the local time when the event or the rearm event is observed. **ERRM_TIME** is the environment variable that the ERRM supplies. The value of **ERRM_TIME** is localized and converted to readable form (*hh:mm:ss mm/dd/yy*) before it is displayed.

This script captures the values of the environment variables and uses the **gless** command to write a message to the specified X-window display.

Parameters

x-display
: Specifies the X-window display to which the event or rearm event is written.

Flags

-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 host 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/displayevent

Examples

1. Suppose the command **/usr/sbin/rsct/bin/displayevent adminhost:0** is an 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 that is defined for this condition is met, and an event occurs. The critical-notification response takes place, and **displayevent** is run. The following message appears on the X-window display of the node **adminhost**:

```
Critical event occurred:
  Condition:  /var space used
  Node:  localnode
  Resource:  /var
  Resource Class:  IBM.FileSystem
  Resource Attribute:  PercentTotUsed
  Attribute Type:  Int32
  Attribute Value:  91
  Time:  12:21:25 02/27/04
```

Related information:

ERRM environment variables

elogevent Command
Purpose

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

Syntax

elogevent [**-h**] *log_file*

Description

The **elogevent** captures event information that is posted by the event response resource manager (ERRM) in environment variables the ERRM generates when an event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. This script always return messages in English.

Event information that is returned about the ERRM environment variables includes the following:

Local Time
: Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script uses the **alog** command to write event information to and read event information from the specified *log_file*.

Flags

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

Parameters

log_file Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 The script has run successfully.

1 A required *log_file* is not specified.

2 The *log_file* path is not valid.

Restrictions

- This script must be run on the node where the ERRM is running.
- The user who runs this script must have write permission for the *log_file* where the event information is logged.

Standard Output

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

Examples

1. To log information, specify the log file as `/tmp/event.log`. ERRM runs this command:
   ```
   /usr/sbin/rsct/bin/elogevent/tmp/event.log
   ```

   The **/tmp/event.log** file does not need to exist when the command is run.
2. To see the contents of the **/tmp/event.log** file, run this command:
   ```
   alog -f /tmp/event.log -o
   ```

   The following sample output shows a warning event for the **/var** file system (a file system resource):
   ```
   ============================================================================
   Event reported at Mon Mar 27 16:38:03 2007

   Condition Name:                         /var space used
   Severity:                               Warning
   Event Type:                             Event
   Expression:                             PercentTotUsed>90

   Resource Name:                          /var
   Resource Class Name:                    IBM.FileSystem
   Data Type:                              CT_UINT32
   Data Value:                             91
   ```

Location

/usr/sbin/rsct/bin/elogevent

logevent Command

Purpose

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

Syntax

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

Description

The **logevent** captures event information that is posted by the event response resource manager (ERRM) in environment variables the ERRM generates when an event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. The language in which the messages of the **logevent** script are returned depend on the locale settings.

Event information that is returned about the ERRM environment variables includes the following:

Local Time
Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script uses the **alog** command to write event information to and read event information from the specified *log_file*.

Flags

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

Parameters

log_file Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 The script has run successfully.

1 A required *log_file* is not specified.

2 The *log_file* path is not valid.

Restrictions

- This script must be run on the node where the ERRM is running.
- The user who runs this script must have write permission for the *log_file* where the event information is logged.

Standard Output

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

Examples

1. To log information, specify **/tmp/event.log** as follows:

   ```
   /usr/sbin/rsct/bin/logevent
   /tmp/event.log
   ```

   The **/tmp/event.log** file does not need to exist when the command is run.
2. To see the contents of the **/tmp/event.log** file, run this command:

   ```
   alog -f /tmp/event.log -o
   ```

The following sample output shows a warning event for the **/var** file system (a file system resource):

```
===============================================================================
Event reported at Mon Mar 27 16:38:03 2007

Condition Name:                         /var space used
Severity:                               Warning
Event Type:                             Event
Expression:                             PercentTotUsed>90

Resource Name:                          /var
Resource Class Name:                    IBM.FileSystem
Data Type:                              CT_UINT32
Data Value:                             91
```

Location

/usr/sbin/rsct/bin/logevent

enotifyevent Command, notifyevent Command

Purpose

Mails event information generated by the event response resource manager (ERRM) 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 depend on the locale settings.

These scripts capture event information that is posted by the event response resource manager (ERRM) in environment variables that are generated by the ERRM when an event occurs. These scripts can be used as actions that are run by an event response resource. They can also be used as templates to create other user-defined actions.

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
: Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

In AIX, 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.

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.

For AIX, 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.

For other platforms, the size of the *log_file* is not limited, and it will not overwrite itself. The file size will increase indefinitely unless the administrator periodically removes entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 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.

Standard Output

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

Examples

1. 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 Sun Mar 26 16:38:03 2002

Condition Name:      /var space used
Severity:     Warning
Event Type:      Event
Expression:      PercentTotUsed>90

Resource Name:     /var
Resource Class Name:    IBM.FileSystem
Data Type:     CT_UINT32
Data Value:     91
```

Location

/usr/sbin/rsct/bin/enotifyevent
Contains the **enotifyevent** script

/usr/sbin/rsct/bin/notifyevent
Contains the **notifyevent** script

ewallevent Command
Purpose

Broadcasts an event or a rearm event to all users who are logged in.

Syntax

ewallevent [**-c**] [**-h**]

Description

The **ewallevent** script broadcasts a message on 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. Event or rearm event information is captured and posted by the event response resource manager in environment variables that are generated by the event response resource manager when an event or a rearm event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. This script always returns messages in English.

Messages are displayed in this format at the consoles of all users who are logged in when an event or a rearm event occurs for which this 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}.
```

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
: Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script captures the environment variable values and uses the **wall** command to write a message to the currently logged-in user consoles.

Flags

-c Instructs **ewallevent** to broadcast the **ERRM_VALUE** of an ERRM event. When the **-c** flag is specified, **ewallevent** broadcasts the SNMP trap message.

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

Parameters

log_file Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 Script has run successfully.

1 Error occurred when the script was run.

Restrictions

1. This script 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.

Standard Output

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

Examples

1. Suppose the **ewallevent** 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 **ewallevent** 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}.
   ```

Location

/usr/sbin/rsct/bin/ewallevent

wallevent Command

Purpose

Broadcasts an event or a rearm event to all users who are logged in.

Syntax

wallevent [-c] [-h]

Description

The **wallevent** script broadcasts a message on 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. Event or rearm event information is captured and posted by the event response resource manager in environment variables that are generated by the event response resource manager when an event or a rearm event occurs. This script can be used as an action that is run by an event response resource. It can also be used as a template to create other user-defined actions. The language in which the messages of the **wallevent** script are returned depend on the locale settings.

Messages are displayed in this format at the consoles of all users who are logged in when an event or a rearm event occurs for which this 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}.
```

Event information is returned about the ERRM environment variables, and also includes the following:

Local Time
: Time when the event or rearm event is observed. The actual environment variable supplied by ERRM is ERRM_TIME. This value is localized and converted to readable form before being displayed.

This script captures the environment variable values and uses the **wall** command to write a message to the currently logged-in user consoles.

Flags

-c Instructs **wallevent** to broadcast the **ERRM_VALUE** of an ERRM event. When the **-c** flag is specified, **wallevent** broadcasts the SNMP trap message.

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

Parameters

log_file Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 Script has run successfully.

1 Error occurred when the script was run.

Restrictions

1. This script 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.

Standard Output

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

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}.
```

Location

/usr/sbin/rsct/bin/wallevent

msgevent
Purpose

Sends an event or a rearm event to a specified user's console.

Syntax

msgevent [**-h**] *user* [*tty*]

Description

The **msgevent** script displays a message about an event or a rearm event on the console of a specified user. If *tty* is specified, the message is sent to that *tty*; otherwise, **msgevent** chooses a *tty* that belongs to *user*. 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.

The **msgevent** script can be used as an action that an event-response resource runs. You can also use this 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*.

This script returns event information about the following ERRM environment variables, which are described in *RSCT: Administration Guide*:

```
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
  Time:  local time
```

This example is a template of the message that is displayed on the specified console or *tty* when an event or a rearm event occurs for which **msgevent** is a response action. For the **Time:** field, this script returns the local time when the event or the rearm event is observed. **ERRM_TIME** is the environment variable that the ERRM supplies. The value of **ERRM_TIME** is localized and converted to readable form (*hh:mm:ss mm/dd/yy*) before it is displayed.

The **msgevent** script captures the values of these ERRM environment variables and displays a message on the specified user's console.

Flags

-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 Script has run successfully.

1 Error occurred when script was run.

Restrictions

This script must be run on the host 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/msgevent

Examples

1. Suppose the command **/usr/sbin/rsct/bin/msgevent root** is an 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 **msgevent** is run. This message is displayed on a tty belonging to **root**:

```
Critical event occurred:
  Condition:  /var space used
  Node:  localnode
  Resource:  /var
  Resource Class:  IBM.FileSystem
  Resource Attribute: PercentTotUsed
  Attribute Type: Int32
  Attribute Value: 91
  Time: 18:43:03 03/28/02
```

Related information:

ERRM environment variables

snmpevent Command
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 defined in its configuration file. This script is meant to be called by the predefined ERRM response **Generate SNMP trap**. Event or rearm event information is captured and posted by ERRM in environment variables that are generated when an ERRM event or a rearm event occurs.

The **snmpevent** script can also be used as a template to create other user-defined actions. See the *RSCT Administration Guide* to understand how an event response resource runs an action command.

The following message template is sent as a trap when an event or a rearm event occurs and **snmpevent** is the defined response:

```
[ERRM_COND_SEVERITY] [ERRM_TYPE] occurred:
          Condition: [ ERRM_COND_NAME]
          Node:  [ERRM_NODE_NAME]
          Resource: [ERRM_RSRC_NAME]
```

```
Resource Class:  [ERRM_RSRC_CLASS_NAME]
Resource Attribute: [ERRM_ATTR_NAME]
Attribute Type: [ERRM_DATA_TYPE]
Attribute Value: [ERRM_VALUE]
```

The environment variables have the following definitions:

ERRM_COND_SEVERITY
Specifies the significance of the condition resource that caused the event or rearm event. The valid values are: Critical, Warning, or Informational.

ERRM_TYPE
Specifies the type of event that occurred. The valid values are: event or rearm event.

ERRM_COND_NAME
Specifies the name of the condition resource with the attribute value that changed to cause this event or rearm event.

ERRM_NODE_NAME
Specifies the host name on which this event or rearm event occurred.

ERRM_RSRC_NAME
Specifies the name of the resource with the attribute that changed to cause this event or rearm event.

ERRM_RSRC_CLASS_NAME
Specifies the name of the resource class to which the resource that caused this event or rearm event belongs.

ERRM_ATTR_NAME
Specifies the name of the resource attribute that changed to cause this event or rearm event.

ERRM_DATA_TYPE
Specifies the data type of the resource attribute.

ERRM_VALUE
Specifies the value of the resource attribute that changed to cause this event or rearm event.

The **snmpevent** command captures these environment variable values and formats a generic message that is sent as a trap via a call to the **snmptrap** command.

Flags

-a *host-name*
Specifies the host name of the SNMP agent to which the AIX subagent will connect. By default, the subagent will connect to the SNMP agent running on the local node.

-c Specifies the SNMP community to be used. This can be any string the SNMP agent will accept. The default is **public**.

-h Writes this script's usage statement to standard output.

Parameters

log_file Specifies the name of the file where event information is logged. An absolute path for the *log_file* parameter should be specified.

The *log_file* is treated as a circular log and has a fixed size of 64KB. When *log_file* is full, new entries are written over the oldest existing entries.

If *log_file* already exists, event information is appended to it. If *log_file* does not exist, it is created so that event information can be written to it.

Exit Status

0 The script has run successfully.

1 An error occurred when the script was run.

Restrictions

This script must be run on the node where the ERRM is running.

Standard Output

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

Examples

1. Suppose the command **/usr/sbin/rsct/bin/snmpevent** is an action in the critical-notification response, which is associated with the CSM predefined condition **NodeChanged**. This can be done with the **mkcondresp** command followed by the **startcondresp** command. The **/etc/snmpdv3.conf** file should be configured to where the trap will be sent. In this example, if you want the trap sent to **9.117.16.246**, write the **/etc/snmpdv3.conf** file as follows:

```
VACM_GROUP group1 SNMPv1 public -

VACM_VIEW defaultView        internet                   - included
-VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -

NOTIFY notify1 traptag trap -

#TARGET_ADDRESS Target1 UDP 127.0.0.1       traptag trapparms1 - - -
TARGET_ADDRESS Target1 UDP 9.117.16.246       traptag trapparms1 - - -

TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -

COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -

DEFAULT_SECURITY no-access - -

logging         file=/usr/tmp/snmpdv3.log       enabled
logging         size=0                          level=0

smux            1.3.6.1.4.1.2.3.1.2.1.2         gated_password  # gated

snmpd smuxtimeout=200 #muxatmd
smux 1.3.6.1.4.1.2.3.1.2.3.1.1 muxatmd_password #muxatmd
```

   Then, restart the **snmpd** daemon by first killing the **snmpd** daemon that is currently running and then starting it again:

```
# ps -ef | grep snmpd
    root  4570 12956   1 08:24:32  pts/0  0:00 grep snmpd
    root 13810     1   0 08:11:04      -  0:00 snmpd
# kill -9 13810
# snmpd
```

   Next, change the LParID property of node c175n08 to 12:

```
# chnode c175n08 LParID=12
```

   Now, on the node **9.117.16.158** (the node with the SNMP manager that was specified in the **/etc/snmpdv3.conf** file), the SNMP manager should record something like this:

```
2002-07-15 09:09:25 c174tr1.ppd.pok.ibm.com [9.114.78.17] TRAP, SNMP v1,
community public
        enterprises.ibm Enterprise Specific Trap (1) Uptime: 0:01:45.00
        enterprises.ibm.ibmProd.191.1.6.1.0 = "Informational Event
```

```
occurred. Condition=NodeChanged Node=c174tr1.ppd.pok.ibm.com
Resource=c175n08.ppd.pok.ibm.com Resource Class=Node Resource
Attribute=Changed Attributes Attribute Type=CT_CHAR_PTR_ARRAY Attribute
Val={LParID} "
```

The output varies based on SNMP managers.

Location

/usr/sbin/rsct/bin/snmpevent

Sensor resource manager commands

chsensor Command

Purpose

Changes the attributes of a resource monitoring and control (RMC) sensor.

Syntax

chsensor [**-m**[**-i** *seconds*] [**-a** | **-n** *host1* [**,** *host2* **,** ...] | **-N** { *node_file* | "-" }] [**-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*.

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.

-m Specifies that the resource to be changed is a microsensor resource.

-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 a file or standard input listing the nodes on which the sensor must be removed. This flag is only appropriate in a Cluster Systems Management (CSM) or a peer domain cluster.

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

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

Parameters

sensor_name
Specifies the name of the sensor to change.

attr1=*value1* [*attr2*=*value2* **...**]
Specifies one or more sensor or microsensor attributes and the their new values.

You can change the values of these sensor 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 whether special handling is required for this sensor. You can specify any combination of these values:

0 Indicates that no special handling is required. This is the default.

The sensor command runs at the interval that is defined for *sensor_name*. The **sensor** command does not run when monitoring begins or when the **lssensor** command is run. A sensor command is a command or script that the sensor resource manager runs to set and update a sensor's attribute values.

1 Indicates that the sensor command runs when monitoring begins. The sensor command also runs at the interval that is defined for *sensor_name*. The sensor command does not run when the **lssensor** command is run.

Specifying this value is not recommended, unless you expect the sensor command to run quickly. If the sensor command does not run quickly, it could block other requests to the sensor resource manager. These requests are not processed until the sensor command finishes running.

2 Indicates that output from the command in the **SavedData** field is not saved permanently to **SavedData** persistent resource attributes. If this value is not specified, the sensor resource manager updates data in the registry's resource table whenever the command's standard output contains the line: **SavedData="***any-string***"**.

3 Indicates a combination of values **1** and **2**

4 Indicates that the sensor resource manager runs the command after 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**.

8 Indicates that the sensor resource manager resets the dynamic attribute values after monitoring is stopped.

UserName
Specifies the name of a user whose privileges are 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 are 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.

You can change the values of these microsensor attributes:

Name Specifies the new name of the microsensor. If the new name is a string that contains spaces or special characters, it must be enclosed in quotation marks.

Description
Provides a description of the microsensor and what it is monitoring.

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.

Exit Status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

6 No sensor resources were found.

n Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Examples

1. To 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
   ```
4. To change the **Name** attribute of microsensor **IBM.msensorq** to **IBM.MSensorQ**, enter:
   ```
   chsensor -m IBM.msensorq Name=IBM.MSensorQ
   ```

Location

/usr/sbin/rsct/bin/chsensor

lssensor Command

Purpose

Displays information about sensors and microsensors that are defined to the resource monitoring and control (RMC) subsystem.

Syntax

lssensor [**-m**] [**-a** | **-n** *host1*[,*host2*...] | **-N** { *node_file* "–" }] [**-l** | **-t** | **-d** | **-D** *delimiter*] [**-x**] [**-h**] [**-v** | **-V**] [**-A** | *sensor_name1* [*sensor_name2*...]]

Description

The **lssensor** command displays the attributes of one or more sensors. If you do not specify any *name* parameters, the **lssensor** command lists the names of all of the sensors. Use the **-A** flag to list all of the sensors and all of their attributes and values. Use the **-m** flag to display information about microsensors.

The **lssensor** command displays values for attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. A sensor command is a command or script that the sensor resource manager runs to

set and update a sensor's attribute values. A microsensor module is a loadable module that the microsensor resource manager runs to set and update a microsensor's attribute values.

Use the **-l**, **-t**, **-d**, or **-D** flags to display the output in long format, table format, or delimiter format. The **-x** flag omits headings when any of these flags are used.

The **lssensor** command runs on any node. If you want **lssensor** to run on all of the nodes in a domain, use the **-a** flag. If you want **lssensor** to run on a subset of nodes in a domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

The **lssensor** command lists the following information about defined sensors:

Field	Description
Name	The name of the sensor.
Command	The command that is run to update the sensor attributes
ConfigChanged	Information about changes to access or to persistent attributes
ControlFlags	Indicates whether any special handling is required for this sensor
Description	This field is not used
ErrorExitValue	Indicates how the exit value is interpreted by the sensor resource manager
ErrorMessage	This field is not used
ExitValue	The exit code from the command that is running
Float32	The type **float32** attribute for this sensor resource
Float64	The type **float64** attribute for this sensor resource
Int32	The type **int32** attribute for this sensor resource
Int64	The type **int64** attribute for this sensor resource
MonitorStatus	This attribute is set to 1 when certain sensor attributes are being monitored
NodeNameList	The name of the node where the sensor resource is defined
RefreshInterval	The interval (in seconds) during which the sensor attribute values are updated when the sensor command is run
SavedData	An output string from the sensor command
SD	Contains all dynamic resource attributes except **ConfigChanged**, **Quantum**, and **ExitValue** as its elements
String	The type string attribute for this sensor resource
TimeCommandRun	Indicates the date and time that the sensor command was run
Uint32	The type **uint32** attribute for this sensor resource
Uint64	The type **uint64** attribute for this sensor resource
UserName	The user ID that is used when run the sensor command is run

The **lssensor** command displays the following information about defined microsensors:

Field	Description
Name	The name of the microsensor.
ActivePeerDomain	The peer domain for which information is being displayed.
Arguments	The arguments for this microsensor resource.
ConfigChanged	Information about changes to persistent attributes or to access.
CustomDynamicAttributes	The custom dynamic attributes for this microsensor resource.
Description	Information about the microsensor and what it monitors.
Float32	The type float32 attribute for this microsensor resource.
Float32Array	The type float32 array attribute for this microsensor resource.
Float64	The type float64 attribute for this microsensor resource.
Float64Array	The type float64 array attribute for this microsensor resource.
Int32	The type int32 attribute for this microsensor resource.
Int32Array	The type int32 array attribute for this microsensor resource.
Int64	The type int64 attribute for this microsensor resource.
Int64Array	The type int64 array attribute for this microsensor resource.
LastQueryRC	The return code from the microsensor module from the last time the microsensor was called for an attribute of the microsensor resource.
LastQueryTime	The time of LastQueryRC.
ModuleName	The path name to the loadable microsensor module.
MonitorStatus	This attribute is set to 1 when any of the other microsensor attributes is being monitored.
NodeNameList	The name of the node where this microsensor is defined.
RefreshInterval	The interval (in seconds) during which the microsensor attribute values are updated when the microsensor callback is called.
String	The type string attribute for this microsensor resource.
StringArray	The type string array attribute for this microsensor resource.
UInt32	The type uint32 attribute for this microsensor resource.
UInt32Array	The type uint32 array attribute for this microsensor resource.
UInt64	The type uint64 attribute for this microsensor resource.
UInt64Array	The type uint64 array attribute for this microsensor resource.

Flags

-a Lists sensors that match the specified name on all nodes in the domain. The CT_MANAGEMENT_SCOPE environment variable determines the cluster scope. If CT_MANAGEMENT_SCOPE is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command will run once for the first valid scope found. For example, if both a management domain and a peer domain exist, **lssensor** -a with CT_MANAGEMENT_SCOPE not set will run in the management domain. In this case, to run in the peer domain, set CT_MANAGEMENT_SCOPE to 2.

-A Displays all of the sensors with their attributes and values.

–d Specifies delimiter-formatted output. The default delimiter is a colon (:). Use the **-D** flag if you want to change the default delimiter.

–D *delimiter*
Specifies delimiter-formatted output that uses the specified delimiter. Use this flag to specify something other than the default colon (:). An example is when the data to be displayed contains colons. Use this flag to specify a delimiter of one or more characters.

–l Specifies that the information be displayed in "long" format. Each attribute is displayed on a separate line.

-m Specifies that information about microsensors will be displayed.

-n *host1*[**,***host2***...**]
Specifies the node from which the sensor should be listed. By default, the sensor is listed from the local node. This flag is only appropriate in a management domain or a peer domain.

-N {*node_file* | **"-"**}
Specifies that node names are read from a file or from standard input. Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (#) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "–"** in a management domain or a peer domain to read the node names from standard input.

–t Specifies table format. Each attribute is displayed in a separate column, with one sensor resource per line.

-x Suppresses header printing when **-l**, **-t**, **-d**, or **-D** is specified.

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

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

Parameters

sensor_name1 [*sensor_name2***...**]
Specifies the names of one or more sensors to display.

Security

To display sensor information using this command, you need read permission for the **IBM.Sensor** resource class. To display microsensor information using this command, you need read permission for the **IBM.MicroSensor** resource class. Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* guide for details on the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

6 No sensor resources were found.

n Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP

address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Examples

1. To list the names of all of the sensors, enter:

```
lssensor
```

The output will look like this:

```
sensor1
sensor2
sensor3
```

2. To list the names and attributes of all sensors, enter:

```
lssensor -A
```

The output will look like this:

```
 Name = sensor1
 ActivePeerDomain =
 Command = /usr/local/bin/sensorcmd1
 ConfigChanged = 0
 ControlFlags = 1
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 1.06381e+06
 Float64 = 1.06381e+06
 Int32 = 1063814
 Int64 = 1063814
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData = Last SavedData
 SD = [string from sensor1,1063814,1063814,1063814,1063814,1.06381e+06,1.06381e+06]
 String = string from sensor1
 Uint32 = 1063814
 Uint64 = 1063814
 UserName = root
-------------------
 Name = CFMRootModTime
 ActivePeerDomain =
 Command = /opt/csm/csmbin/mtime/cfmroot
 ConfigChanged = 0
```

```
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
-------------------
 Name = ErrorLogSensor
 ActivePeerDomain =
 Command = /opt/csm/csmbin/monerrorlog
 ConfigChanged = 0
 ControlFlags = 0
 Description =
 ErrorExitValue = 1
 ExitValue = 0
 Float32 = 0
 Float64 = 0
 Int32 = 0
 Int64 = 0
 NodeNameList = {somenode.pok.ibm.com}
 RefreshInterval = 60
 SavedData =
 SD = [,0,0,0,0,0,0]
 String =
 Uint32 = 0
 Uint64 = 0
 UserName = root
-------------------
.
.
.
```

3. To list the attributes of **sensor2**, enter:

```
lssensor sensor2
```

The output will look like this:

```
Name = sensor2
Command = /usr/local/bin/sensorcmd2
ConfigChanged = 0
ControlFlags = 0
Description =
ErrorExitValue = 1
ExitValue = 127
Float32 = 0
Float64 = 0
Int32 = 0
Int64 = 0
NodeNameList = {somenode.pok.ibm.com}
RefreshInterval = 60
SavedData =
SD = [,0,0,0,0,0,0]
String =
Uint32 = 0
Uint64 = 0
UserName = root
```

4. To list all of the sensors' information using delimited output, enter:

```
lssensor -dA
```

The output will look like this:

```
Displaying sensor information:
Name:ActivePeerDomain:Command:ConfigChanged:ControlFlags:Description:ErrorExitValue:ErrorMessage:ExitValue:
Float32:Float64:Int32:Int64:MonitorStatus:NodeNameList:RefreshInterval:SD:SavedData:
String:TimeCommandRun:Uint32:Uint64:UserName:
JoeExample:JoeDomain:cat /etc/motd:0:0::1::0:
::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
:Fri Feb  6 19:00:00 2009:::root:
JoeSample:JoeDomain:/usr/sbin/rsct/install/bin/ctversion:0:0::1::0:
::::0:{node1.myhost.com}:60:[,0,0,0,0,0,0]::
:Fri Feb  6 19:00:00 2009:::root:
JoeSens:JoeDomain:/tmp/sensor/numusers:0:1::1::0:
::::0:{node1.myhost.com}:0:[,2,0,0,0,0,0]::
:Tue Mar  3 10:27:19 2009:::root:
```

5. To list the names of all of the sensors on the nodes that are listed in the **/u/joe/common_nodes** file, enter:

```
lssensor -N /u/joe/common_nodes
```

where /u/joe/common_nodes contains:

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

The output will look like this:

```
sensor1
sensor2
sensor3
```

6. To list the names of all of the microsensors, enter:

```
lssensor -m
```

The output will look like this:

```
IBM.MSensor1
IBM.MSensor2
IBM.MSensor3
```

7. To list the attributes of the microsensor **IBM.MSensor2**, enter:

```
lssensor -m IBM.MSensor2
```

The output will look like this:

```
Name = IBM.MSensor2
ActivePeerDomain =
Arguments = all
ConfigChanged = 0
CustomDynamicAttributes = {[CDA1,19,1,3,0,1],[CDA2,20,2,2,0,1],[CDA3,21,3,2,0,1]
Description =
Float32 =
Float32Array =
Float64 =
Float64Array =
Int32 = 52
Int32Array = {36, 45, 2, 73}
Int64 =
Int64Array =
LastQueryRC = 0
LastQueryTime = Tue Mar 31 18:00:00 2009
ModuleName = /usr/slib/msensors/sensor2
MonitorStatus = 0
NodeNameList = {node2.gumby.com}
RefreshInterval = 600
String =
StringArray =
```

```
UInt32 =
UInt32Array =
UInt64 =
UInt64Array =
```

Location

/usr/sbin/rsct/bin/lssensor

mksensor Command
Purpose

Defines a sensor or a microsensor to the resource monitoring and control (RMC) subsystem.

Syntax

To define a sensor:

mksensor [**–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–i** *seconds*] [**–c** *n*] [**–e 0** | **1** | **2**] [**–u** *user-ID*] [**–h**] [**–v** | **–V**] *sensor_name* [**"**]*sensor_command*[**"**]

To define a microsensor:

mksensor -m [**–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–i** *seconds*] [**–h**] [**–v** | **–V**] *microsensor_name microsensor_module* [[**"**]*microsensor_arguments*[**"**]]

Description

The **mksensor** command defines a sensor resource to the resource monitoring and control (RMC) subsystem. A *sensor* is an RMC resource with attributes that you can monitor. You can use the event-response resource manager (ERRM) commands to set up monitoring of the sensor attributes. The response actions defined will run when a monitored sensor event occurs. This enables administrators to extend RMC monitoring capabilities without having to write a resource manager.

For sensors, the *sensor_command* parameter specifies the command or script that the sensor resource manager will run to set (and then later, update) the sensor attribute values. After the sensor attributes have been monitored, the sensor resource manager sets the attribute values. Then, at defined intervals, the sensor resource manager updates these values.

For microsensors, the *microsensor_module* parameter specifies the path name to the loadable module that the microsensor resource manager will call to set (and then later, update) the microsensor attribute values. After the microsensor attributes have been monitored, the microsensor resource manager sets the attribute values. Then, at defined intervals, the microsensor resource manager updates these values. Use the **-m** flag to create a microsensor.

Alternatively, you can use **chsensor** or **refsensor** to update the sensor or microsensor attribute values. The **lssensor** command displays values for sensor or microsensor attributes that you can set using a sensor command or a microsensor module, if the attributes are monitored. If the attributes are not monitored, **lssensor** does not display their values. To remove a sensor or a microsensor, use the **rmsensor** command.

The **mksensor** command runs on any node. To define a sensor or a microsensor on one or more nodes in a management domain or a peer domain, use the **-n** flag. Instead of specifying multiple node names using the **-n** flag, you can use the **-N** *node_file* flag to indicate that the node names are in a file. Use **-N "–"** to read the node names from standard input.

If Cluster Systems Management (CSM) is installed on your system, you can use CSM defined node groups as node name values to refer to more than one node. For information about working with CSM node groups and using the CSM **nodegrp** command, see the *CSM: Administration Guide* and the *CSM: Command and Technical Reference*.

A sensor consists of the following attributes, which can be set using a sensor command: :

Float32
The type float32 attribute for this sensor resource.

Float64
The type float64 attribute for this sensor resource.

Int32 The type int32 attribute for this sensor resource.

Int64 The type int64 attribute for this sensor resource.

Quantum
The type quantum attribute for this sensor resource.

String The type string attribute for this sensor resource.

Uint32
The type uint32 attribute for this sensor resource.

Uint64
The type uint64 attribute for this sensor resource.

A sensor command sets attribute values by sending the values to standard output in a format that the sensor resource manager can parse. The format is *attr=value*. For example, if the sensor command sets the **Int32** attribute to 57, it writes **Int32=57** to standard output. To set more than one attribute value, the sensor command can write multiple *attr=value* pairs to standard output. The *attr=value* pairs can be on one or more lines. If the sensor command output is not in *attr=value* form, it is assumed to be a string and the value is placed in the **String** attribute.

The sensor command runs using the user ID that creates the sensor resource. Once a sensor resource is monitored, the sensor command is run at intervals specified by the **-i** flag, which is expressed in seconds. The default interval is 60 seconds. Specify a value of 0 to indicate that the sensor command is not to run at intervals. In this case, the **refsensor** command is typically used to update the sensor values.

Use the **-e** flag to control how the exit values from *sensor_command* are interpreted. Depending on this setting, when the exit value of the *sensor_command* is considered to be an error, the sensor attributes are not set and information is written to the audit log.

A microsensor consists of the following attributes, which can be set using a microsensor load module:

Float32
The type **float32** attribute for this microsensor resource

Float32Array
The type **float32** array attribute for this microsensor resource

Float64
The type **float64** attribute for this microsensor resource

Float64Array
The type **float64** array attribute for this microsensor resource

Int32 The type **int32** attribute for this microsensor resource

Int32Array
The type **int32** array attribute for this microsensor resource

Int64 The type **int64** attribute for this microsensor resource

Int64Array
The type **int64** array attribute for this microsensor resource

Quantum
The type **quantum** attribute for this microsensor resource.

String The type **string** attribute for this microsensor resource.

StringArray
The type **string** array attribute for this microsensor resource.

UInt32
The type **uint32** attribute for this microsensor resource.

UInt32Array
The type **uint32** array attribute for this microsensor resource.

UInt64
The type **uint64** attribute for this microsensor resource.

UInt64Array
The type **uint64** array attribute for this microsensor resource.

The microsensor resource manager will make calls to the microsensor load module to set the values of the microsensor attributes. See the *Administering RSCT* for information about how to use microsensors.

Flags

–m Specifies that the resource to be defined is a microsensor resource.

–n *host1***[,***host2***...]**
Specifies one or more nodes on which the sensor should be defined. By default, the sensor is defined on the local node. This flag is only appropriate in a management domain or a peer domain.

–N { *node_file* **| "–" }**
Specifies that node names are read from a file or from standard input.

Use **-N** *node_file* to indicate that the node names are in a file.

- There is one node name per line in *node_file*
- A number sign (**#**) in column 1 indicates that the line is a comment
- Any blank characters to the left of a node name are ignored
- Any characters to the right of a node name are ignored

Use **-N "–"** in a management domain or a peer domain to read the node names from standard input.

–i *seconds*
Specifies the interval at which a sensor command is run to update a sensor's attribute values or a microsensor module is run to update a microsensor's attribute valuesyup. *seconds*, which is an integer value, must be greater than or equal to **10**. The default interval is **60** seconds.

The sensor command is run at the specified interval only when the sensor resource is monitored. The microsensor module is run at the specified interval only when the microsensor resource is monitored. If the interval is set to **0**, the sensor command or microsensor module will not run automatically.a

Using this flag is independent of using the **refsensor** command to refresh a sensor.

–c *n* Specifies whether special handling is required for this sensor. *n* can be one of these values:

0 Indicates that no special handling is required. This is the default.

The sensor command will run at the interval that is defined for *sensor_name*. The sensor command will *not* run when monitoring begins or when the **lssensor** command is run.

1 Indicates that the sensor command will run when monitoring begins. The sensor command will also run at the interval that is defined for *sensor_name*. The sensor command will *not* run when the **lssensor** command is run.

Specfiying this value is not recommended, unless you expect the sensor command to run quickly. If the sensor command does not run quickly, it could block other requests to the sensor resource manager. These requests will not be processed until the sensor command finishes running.

2 Indicates that output from the command in the **SavedData** field is not saved permanently to **SavedData** persistent resource attributes. If this value is not specified, the sensor resource manager updates data in the registry's resource table whenever the command's standard output contains the line: **SavedData="***any-string***"**.

3 Indicates a combination of values **1** and **2**.

4 Indicates that the sensor resource manager will run the sensor command after monitoring has stopped.

5 Indicates a combination of values **1** and **4**.

6 Indicates a combination of values **2** and **4**.

7 Indicates a combination of values **1**, **2**, and **4**.

–e 0 | 1 | 2
Specifies how the sensor resource manager interprets the exit values of *sensor_command*, as follows:

0 No exit value from *sensor_command* is an error.

1 An exit value other than **0** from *sensor_command* is an error.

2 An exit value of **0** from *sensor_command* is an error.

The default value is **1**. The sensor attributes are not updated when the exit value is interpreted as an error. For an error, information is written to the audit log.

–u *user-ID*
Specifies the name of a user whose privileges will be used to run the sensor command. The user should already be defined on the system. The default value for *user-ID* is the user name that is associated with the current effective user ID.

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

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

Parameters

["]*microsensor_argument***["]**
Specifies a string that will be passed to the microsensor module callback function. The microsensor resource manager will break the string into an array of strings based on blank characters in the microsensor argument. The microsensor argument cannot be changed once the microsensor is defined.

If the microsensor argument contains any blank characters or any special characters that can be interepreted by the shell, it must be enclosed in double quotation marks. When the microsensor argument is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**).

microsensor_module
Specifies the path name to the loadable microsensor module. A signature for the module is stored by the microsensor resource manager and is verified when the module is used. The microsensor module cannot be changed once the microsensor is defined.

microsensor_name
Specifies the name of the microsensor that is to be defined.

["]*sensor_command***["]**
Specifies a command or script that the sensor resource manager will use to set the attribute values of the sensor. You should not call any of the sensor resource manager commands (**chsensor**, **lssensor**, **mksensor**, **refsensor**, or **rmsensor**) as part of this parameter.

If *sensor_command* contains any blank characters, or any special characters that can be interpreted by the shell, it must be enclosed in double quotation marks.

When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**). See Example 2 for more information.

sensor_name
Specifies the name of the sensor that is to be defined.

Security

To create sensors using this command, you need write permission for the **IBM.Sensor** resource class.

To create microsensors using this command, you need write permission for the **IBM.MicroSensor** resource class.

Permissions are specified in the access control list (ACL) file on the contacted system. See the *Administering RSCT* for details on the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

n Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Restrictions

You should not call any of the sensor resource manager commands (**chsensor**, **lssensor**, **mksensor**, **refsensor**, or **rmsensor**) as part of the *sensor_command* parameter, as this could cause a deadlock.

Implementation Specifics

This command is part of the **rsct** fileset for the AIX operating system and **rsct-3.1.0.0-0.***platform***.rpm** package for the Linux, Solaris, and Windows platforms, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Examples

1. To create a new sensor called **Sensor1** that runs the script **/usr/bin/updateSensor1**, which will update the sensor attributes every 30 seconds (once monitored), enter:

   ```
   mksensor -i 30 Sensor1 "/usr/bin/updateSensor1"
   ```

   The contents of **/usr/bin/updateSensor1** may be like:

   ```
   #!/usr/bin/perl
   my $int32 = some_fn_that_generates_i32_value;
   my $string = some_fn_that_generates_string_value;
   print "Int32=$int32 String=$string";
   exit 0;
   ```

   A sample condition could be:

   ```
   mkcondition -r IBM.Sensor -s "Name==Sensor1" -e "Int32 > 100" Sensor1Int32
   ```

   Using the response "E-mail root anytime", a start monitoring command may be:

   ```
   startcondresp Sensor1Int32 "E-mail root anytime"
   ```

2. To create a sensor called **Sensor1** with a *sensor_command* value of

   df -m /var | sed '1d' | sed 's/%//g' | /bin/awk '{ print "Int32="$4}',

   enter:

   ```
   mksensor Sensor1 "df -m /var | sed '1d' | sed 's/%//g' | /bin/awk \
   '{ print \"Int32=\"\$4}'"
   ```

   When *sensor_command* is enclosed in double quotation marks, you must include a backslash escape character (\) before an "inner" double quotation mark. You must also include a \ before a dollar sign (**$**). So in this example, the sensor command substring **"Int32="$4** becomes **\"Int32=\"\$4** when it is part of **mksensor** command.

3. To create a sensor called **Sensor3** that runs the **/usr/bin/checkhealth** script on the nodes that are listed in the **/u/joe/common_nodes** file, enter:

   ```
   mksensor -N /u/joe/common_nodes Sensor3 "/usr/bin/checkhealth"
   ```

   where **/u/joe/common_nodes** contains:

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

4. To create a microsensor called **IBM.msensorq** that uses the shared module **/usr/lib/msensors/msensorq** and requires the parameters **db=abc**, **confirm=yes**, **retry=yes**, and **mirror=no**, enter:

```
mksensor -m IBM.msensorq /usr/lib/msensors/msensorq \
"db=abc confirm=yes retry=yes mirror=no"
```

Location

/usr/sbin/rsct/bin/mksensor

refsensor Command

Purpose

Refreshes a sensor or a microsensor defined to the resource monitoring and control (RMC) subsystem.

Syntax

To refresh a sensor:

refsensor [**–a** | **–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–h**] [**–v** | **–V**] *sensor_name*

To refresh a microsensor:

refsensor –m [**–a** | **–n** *host1*[**,***host2*...] | **–N** { *node_file* | **"–"** }] [**–h**] [**–v** | **–V**] *sensor_name*

Description

The **refsensor** command refreshes a sensor or microsensor resource that is defined to the RMC subsystem. *Sensors* and *microsensors* are RMC resources with attributes that can be monitored. Sensors and microsensors must be monitored for **refsensor** to run successfully.

A 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. A microsensor can be refreshed using **refsensor** to query the values of the microsensor's load module. Use the **-m** flag to refresh a microsensor.

When the **refsensor** command runs, it does not affect the interval, if any, that is defined (using **mksensor**) for running the sensor command or for querying the microsensor load module . That is, if a monitored sensor or microsensor is being updated every 60 seconds, running **refsensor** does not cause the interval timer to be reset to 60 seconds.

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 **Int32** and **Float32** sensor attributes for the sensor named **Sensor1**, enter:

```
refsensor Sensor1 Int32=45 Float32=7.8
```

Microsensor attributes cannot be updated separately.

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.

–m Specifies that the resource to be refreshed is a microsensor resource.

–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.

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.

Security

To refresh sensors using this command, you need write permission for the **IBM.Sensor** resource class.

To refresh microsensors using this command, you need write permission for the **IBM.MicroSensor** resource class.

Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

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.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the **rsct.core** fileset for AIX and **rsct.core-3.1.0.0-0.***platform***.rpm** package for Linux, Solaris, and Windows, where *platform* is **i386**, **ppc**, **ppc64**, **s390**, or **x86_64**.

Examples

1. To refresh the sensor called **Sensor1** so that its defined sensor command is run, enter:

   ```
   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"**, enter:

   ```
   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, enter:

   ```
   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
   ```

4. To refresh the microsensor called **IBM.Sensor1** so that the attribute values are queried using the defined microsensor load module, enter:

   ```
   refsensor -m IBM.Sensor1
   ```

Location

/usr/sbin/rsct/bin/refsensor

rmsensor Command
Purpose

Removes a sensor or a microsensor from the resource monitoring and control (RMC) subsystem.

Syntax

rmsensor [**-m**] [**-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 **IBM.Sensor** resource class or one or more microsensors from the **IBM.MicroSensor** resource class in the RMC subsystem. Use the **-m** flag to remove a microsensor.

If the sensor or microsensor is being monitored, monitoring will be stopped, but the event response resource manager (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 or microsensor.

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*.

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.

-m Specifies that the resources to be removed are microsensor resources.

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

-n *host1***[,***host2***...]**
Specifies the node 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 a file or standard input listing the nodes on which the sensor must be removed. This flag is only appropriate in a Cluster Systems Management (CSM) or a peer domain cluster.

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

Parameters

sensor_name1 **[***sensor_name2***...]**
Specifies one or more names of sensors to remove.

Security

To remove sensors using this command, you need write permission for the **IBM.Sensor** resource class. To remove microsensors using this command, you need write permission for the **IBM.MicroSensor** resource class. Permissions are specified in the access control list (ACL) file on the contacted system. See the *RSCT: Administration Guide* for details on the ACL file and how to modify it.

Exit Status

0 The command has run successfully.

1 An incorrect combination of flags and parameters has been entered.

6 No sensor resources were found.

n Based on other errors that can be returned by the RMC subsystem.

Environment Variables

CT_CONTACT
When the **CT_CONTACT** environment variable is set to a host name or IP address, the command contacts the resource monitoring and control (RMC) daemon on the specified host. If this environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The resource class or resources that are displayed or modified by the command are located on the system to which the connection is established.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based

network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the the session with the RMC daemon to monitor and control the resources and resource classes. The management scope determines the set of possible target nodes where the resources and resource classes can be monitored and controlled.

The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

This command is part of the rsct.core fileset for AIX®.

Examples

1. To remove the sensor **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
   ```
3. To remove the microsensor called **IBM.usensor1**, enter:
   ```
   rmsensor -m IBM.usensor1
   ```

Location

/usr/sbin/rsct/bin/rmsensor

Audit log resource manager commands

lsaudrec Command

Purpose

Lists records from the audit log.

Syntax

lsaudrec [**-l**] [**-a** | **-n** *node_name1*[**,***node_name2*]...] [**-S** *subsystem_name*] [**-s** *selection_string*] [**-x**] [**-h**] [*field_name1* [*field_name2*...]]

Description

The **lsaudrec** command is used to list records in the audit log. The audit log is a facility for recording information about the system's operation. It can include information about the normal operation of the system as well as failures and other errors. It augments the error log functionality by conveying the relationship of the error relative to other system activities. All detailed information about failures is still written to the AIX® error log.

Records are created in the audit log by subsystems that have been instrumented to do that. For example, the event response subsystem runs in the background to monitor administrator-defined conditions and then invokes one or more actions when a condition becomes true. Because this subsystem runs in the background, it is difficult for the operator or administrator to understand the total set of events that occurred and the results of any actions that were taken in response to an event. Because the event response subsystem records its activity in the audit log, the administrator can easily view its activity as well as that of other subsystems using this command.

Each record in the audit log contains named fields. Each field contains a value that provides information about the situation corresponding to the record. For example, the field named **Time** indicates the time at which the situation occurred. Each record has a set of common fields and a set of subsystem-specific fields. The common fields are present in every record in the audit log. The subsystem-specific fields vary from record to record. Their names are only significant when used with a subsystem name because they may not be unique across all subsystems. Each record is derived from a template that defines which subsystem-specific fields are present in the record and defines a format string that is used to generate a message describing the situation. The format string may use record fields as inserts. A subsystem typically has many templates.

The field names can be used as variables in a *selection string* to choose which records are displayed. A selection string is an expression that is made up of field names, constants, and operators. The syntax of a selection string is similar to an expression in the C programming language or the SQL "where" clause. The selection string is matched against each record using the referenced fields of each record to perform the match. Any records that match are displayed. The selection string is specified with the **-s** flag. For information on how to specify selection strings, see the *Administering RSCT* guide.

You can also specify field names as parameters to this command to choose which fields are displayed and the order in which they are displayed. The common field names are:

Field	Description
Time	The time when the situation occurred that the record corresponds to. The value is a 64-bit integer and represents the number of microseconds since UNIX Epoch (00:00:00 GMT January 1, 1970). See the constants below for specifying the time in more user-friendly formats.
Subsystem	The subsystem that generated the record. This is a string.
Category	Indicates the importance of the situation corresponding to the audit record, as determined by the subsystem that generated the record. The valid values are: **0** (informational) and **1** (error).
SequenceNumber	The unique 64-bit integer that is assigned to the record. No other record in the audit log will have the same sequence number.
TemplateId	The subsystem-dependent identifier that is assigned to records that have the same content and format string. This value is a 32-bit unsigned integer.
NodeName	The name of the node from which the record was obtained. This field name cannot be used in a selection string.

In addition to the constants in expressions, you can use the following syntax for dates and times with this command:

#*mmddhhmmyyyy*

This format consists of a sequence of decimal characters that are interpreted according to the pattern shown. The fields in the pattern are, from left to right: *mm* = month, *dd* = day, *hh* = hour,

mm = minutes, *yyyy* = year. For example, **#010523042004** corresponds to January 5, 11:04 PM, 2004. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 0, hour = 0, day = 1, and month = the current month.

#-*mmddhhmmyyyy*
This format is similar to the previous one, but is relative to the current time and date. For example, the value **#-0001** corresponds to one day ago and the value **#-010001** corresponds to one month and one hour ago. Fields can be omitted starting from the right and are replaced by 0.

The audit records considered for display and matched against the selection string can be restricted to a specific subsystem by using the **-S** flag. If this flag is specified, the subsystem-specific field names can be used in the selection string in addition to the common field names.

The nodes from which audit log records are considered for display and matched against the selection string can be restricted to a set of specific nodes by using the **-n** flag. If this flag is specified, the search is limited to the set of nodes listed. Otherwise, the search is performed for all nodes defined within the current management scope, as determined by the CT_MANAGEMENT_SCOPE environment variable.

The audit records are displayed in a table. Field names specified as parameters control which fields are displayed and the order in which they appear on each line. By default, the columns displayed are: the date and time, the subsystem name that generated the record, the severity of the situation, and the subsystem-specific message that describes the situation. If the management scope is not local, the node name is displayed in the first column.

Flags

-l Indicates that long output should be produced. Long output includes subsystem-specific fields that are not included in the formatted message text.

-a Specifies that records from all nodes in the domain are to be displayed. If both the **-n** and the **-a** flags are omitted, records from the local node only are displayed.

-n *node_name1*[**,***node_name2*]**...**
Specifies the list of nodes containing audit log records that will be examined and displayed if they meet the other criteria, such as matching the specified selection string. Node group names can also be specified, which are expanded into a list of node names. If both the **-n** and the **-a** flags are omitted, records from the local node only are displayed.

-S *subsystem_name*
Specifies a subsystem name. If this flag is present, only records identified by *subsystem_name* are considered for display. The records displayed can be further restricted by the **-s** flag. If the subsystem name contains any spaces, it must be enclosed in single or double quotation marks.

For backward compatibility, the subsystem name can be specified using the **-n** flag *only* if the **-a** and the **-S** flags are *not* specified.

-s *selection_string*
Specifies a selection string. This string is evaluated against each record in the audit log. All records that match the selection string will be displayed. If the selection string contains any spaces, it must be enclosed in single or double quotation marks. For information on how to specify selection strings, see the *Administering RSCT* guide.

The names of fields in the record can be used in the expression. If the **-S** flag is not specified, only the names of common fields can be used. See the **Description** for a list of the common field names and their data types. If the **-S** flag is specified, the name of any field for the specified subsystem as well as the common field names can be used.

If this flag is omitted, the records that are displayed will depend on the **-S** flag. If the **-S** flag is omitted, all records from the audit log are displayed. Otherwise, all records for the subsystem identified by the **-S** flag are displayed.

-x Excludes the header (suppresses header printing).

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

Parameters

field_name1 **[***field_name2***...]**
Specifies one or more fields in the audit log records to be displayed. The order of the field names on the command line corresponds to the order in which they are displayed. If no field names are specified, **Time**, **Subsystem**, **Severity**, and **Message** are displayed by default. If the management scope is not local, **NodeName** is displayed as the first column by default. See the **Description** for information about these and other fields.

Security

In order to list records from an audit log when the **-S** flag is omitted, you must have read access to the target resource class on each node from which records are to be listed. When the **-S** flag is specified, you must have read access to the audit log resource corresponding to the subsystem identified by the **-S** flag on each node from which records are to be listed.

Authorization is controlled by the RMC access control list (ACL) file that exists on each node.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon is established. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that can be affected by this command.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines (in conjunction with the **-a** and **-n** flags) the management scope that is used for the session with the RMC daemon. The management scope determines the set of possible target nodes where audit log records can be listed. If the **-a** and **-n** flags are not specified, local scope is used. When either of these flags is specified, CT_MANAGEMENT_SCOPE is used to determine the management scope directly. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is *not* set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Examples

1. To list all records in the audit log on every node in the current management scope as determined by the CT_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 -SERRM -s"ConditionName=='Condition1' && Etype==92"
   ```
8. To list the sensor resource manager records in the audit log on the local node, enter:
   ```
   lsaudrec -SSSRM
   ```
   The output will look like this:

```
Time                Subsystem   Category    Description
11/10/05 21:52:32   SSRM        Error       The Command /SENSOR/sensor.ksh 1
                                            in Sensor SENSOR_NOUSER_1 execution fails.
11/10/05 21:52:36   SSRM        Error       The Command /SENSOR/sensor.nocmd 1 in Sensor
                                            SENSOR_NOCMD_1 exits with error 127.
```

9. To list, in long format, the sensor resource manager records in the audit log on the local node, enter:
   ```
   lsaudrec -l -SSSRM
   ```
   The output will look like this:

```
Time        = 11/10/05 21:52:32 243097
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.ksh 1 in Sensor SENSOR_NOUSER_1 execution fails.
ErrorMsg    = 2645-202 The user name "guest" that was specified for running the command does not exist.

Time        = 11/10/05 21:52:36 361726
```

```
Subsystem   = SSRM
Category    = Error
Description = The Command /SENSOR/sensor.nocmd 1 in Sensor SENSOR_NOCMD_1 exits with error 127.
StandardOut =
StandardErr = ksh: /u/diane/drmc/scripts/SENSOR/sensor.nocmd:  not found
```

10. To list error records only, enter:

```
lsaudrec -s"Category=1"
```

Location

/usr/sbin/rsct/bin/lsaudrec

rmaudrec Command
Purpose

Removes records from the audit log.

Syntax

rmaudrec [**-a** | **-n** *node_name1*[**,***node_name2*]...] [**-S** *subsystem_name*]
-s *selection_string* [**-h**] [**-V**]

Description

The **rmaudrec** command is used 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 *Administering RSCT* 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, 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 the *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.

Parameters

field_name1 **[***field_name2***...]**
Specifies one or more fields in the audit log records to be displayed. The order of the field names on the command line corresponds to the order in which they are displayed. If no field names are specified, **Time**, **Subsystem**, **Severity**, and **Message** are displayed by default. If the management scope is not local, **NodeName** is displayed as the first column by default. See the **Description** for information about these and other fields.

Security

In order to remove records from an audit log when the **-S** flag is omitted, a user 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, the user 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.

Exit Status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with a command-line interface script.

3 An incorrect flag was entered on the command line.

4 An incorrect parameter was entered on the command line.

5 An error occurred that was based on incorrect command-line input.

Environment Variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon is established. When CT_CONTACT is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If CT_CONTACT is not set, the command

contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that can be affected by this command.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT only has meaning if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines (in conjunction with the **-a** and **-n** flags) the management scope that is used for the session with the RMC daemon. The management scope determines the set of possible target nodes where audit log records can be 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.

Implementation Specifics

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

Standard Output

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

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.

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
   ```

Location

/usr/sbin/rsct/bin/rmaudrec

Cluster security services commands

RSCT applications and components use cluster security services to perform authentication within both management and peer domains. The following commands are used to administer tasks related to cluster security services.

ctaclfck Command

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 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.

Entry Type	Description
r	• Format: 0x1 • Permission: read • Operation: generic read operation
w	• Format: 0x2 • Permission: write • Operation: generic write operation
c	• Format: 0x4 • Permission: control • Operation: generic control operation or RMC refresh configuration operation
x	• Format: 0x8 • Permission: run • Operation: generic execute operation
C	• Format: 0x10 • Permission: cancel • Operation: generic cancel operation
q	• Format: 0x20 • Permission: query • Operation: RMC query resource operation
l	• Format: 0x40 • Permission: list • Operation: RMC enumerated resources operation
e	• Format: 0x80 • Permission: event • Operation: RMC event registration, unregistration, and querying
d	• Format: 0x100 • Permission: define • Operation: RMC define and undefine resource operation
v	• Format: 0x200 • Permission: validate • Operation: RMC validate resource handle operation
s	• Format: 0x400 • Permission: set • Operation: 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.

Security

The file system permission of the ACL file is determined by the end user or the application owning 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.

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
   ```

Location

/usr/sbin/rsct/bin/ctaclfck
Contains the **ctaclfck** command

ctcasd Daemon
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 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

through a network connection. **ctcasd** is not used when service requesters and providers establish a secured execution environment through a local operating system connection such as a UNIX domain socket.

When a service requester and a service provider have agreed to use HBA authentication through the cluster security services, the cluster security services library uses **ctcasd** to obtain and authenticate HBA credentials. Cluster security services does 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 both the node's public and private key files are not present, **ctcasd** assumes that it is being started for the first time and creates these files. 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 AF_INET-configured and active 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** creates new keys for the node that do not match the keys stored on the other cluster nodes. If RSCT HBA and HBA2 authentications 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. In Linux-based clusters, records are created in 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).

Restrictions

- The **ctcasd** daemon does not encrypt the HBA 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
Contains the **ctcasd** daemon

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

ctmsskf Command

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.

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.

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.

Restrictions

This command works only on MSS-formatted key files.

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.

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
   ```

Location

/usr/sbin/rsct/bin/ctmsskf
Contains the **ctmsskf** command

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

ctscachgen Command

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 HBA 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.

Security

Permissions on the **ctscachgen** command permit only **root** to run the command.

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.

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 protections 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 generated by this command are generated 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).

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.

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
   ```

Location

/usr/sbin/rsct/bin/ctscachgen
Contains the **ctscachgen** command

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

ctscfg Command

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 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 choose 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 functions 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 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 virtual memory of the process.

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 must 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 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 usage statement to standard output.

Standard output

When the **-h** flag is specified, this command 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 content is not valid or is corrupted.

30 Lock error. An error occurred during the locking of the MAL configuration file.

36 Permission error. The invoker does not have permission to list or modify the MAL configuration file.

105 File error. An error occurred during the reading or writing of the MAL configuration file.

Files

/var/ct/cfg/ctsec.cfg
Working copy of the MAL configuration file

/var/ct/cfg/ctsec.cfg.bak
Backup of the working copy of the MAL configuration file

Security

This command lists and modifies the MAL configuration file. The default version of the MAL configuration file that is installed by RSCT is protected using the file system 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 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
   ```

ctsidmck Command

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 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 authorization functions such 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.

This command allows an administrator to 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 system 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 system, and what entry was used from the identity mapping definition files to obtain this mapping.

Flags

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

-i Displays a list of the supported security mechanisms on this system. 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 system supports.

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.

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.

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 system. **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 The **ctsidmck** command cannott 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.

Restrictions

This command works only on MSS-formatted key files.

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.

Examples

1. To get a list of the security mechanisms that the local system supports, before verifying an identity map, enter:

   ```
   ctsidmck -i
   ```

2. To get only the mapped identity for the RSCT host-based authentication (HBA) mechanism security 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 HBA mechanism's security network identity **glorfindel@rivendell.elvin.net@endor**, enter:

   ```
   ctsidmck -d h -m unix glorfindel@rivendell.elvin.net@endor
   ```

Location

/usr/sbin/rsct/bin/ctsidmck
Contains the **ctsidmck** command

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.

ctskeygen Command

Purpose

Generates cluster security services private and public keys for the local system 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). 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 HBA authentication.

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 HBA authentications 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 system.

-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.

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.

Security

Permissions on the **ctskeygen** command permit only **root** to run the command.

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.

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.

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.

Examples

1. To obtain the list of supported key generation methods:
   ```
   ctskeygen -i
   ```
2. To create new host identifier keys for the local system using the default settings:
   ```
   ctskeygen -n
   ```
3. To create new host identifier keys for the local system 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
   ```

Location

/usr/sbin/rsct/bin/ctskeygen
Contains the **ctskeygen** command

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.

ctsthl Command

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.ibm.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 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 system.

-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, In AIX, host identifier keys will be much longer than this example, making it too large for the command line limit on some 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 system for all known IP addresses and host names associated with AF_INET-configured and active adapters that the daemon can detect. For any host name or IP address on the local system that is not found in the local system's trusted host list file, an entry is added to associate that value with the local system's public key value.

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.

Security

Permissions on the **ctsthl** command permit only **root** to run the command.

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 (AF_INET) interfaces for the local system while processing the **-s** flag. The local system's identities may not be properly

recorded to the trusted host list. Verify that at least one AF_INET or AF_INET6 interface is defined and active on the local system 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.

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.

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.

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 system **zathras.ibm.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 system **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
   ```

Location

/usr/sbin/rsct/bin/ctsthl
Contains the **ctsthl** command

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.

ctstrtcasd Utility

Purpose

Serves as the launch utility of the **ctcasd** daemon for the cluster security services.

Syntax

ctstrtcasd [**-a**] [**-v**]

Description

The **ctstrtcasd** utility is started by the cluster security services to start the **ctcasd** daemon. This utility is provided as a set-user-identity-on-execution binary file, providing the clients of cluster security services 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 agree 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 started by user applications.

The **ctcasd** daemon 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 be activated only 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 start the **ctstrtcasd** utility to start the **ctcas** subsystem if the service is not active. This utility temporarily grants the clients of cluster security services 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 the **ctstrtcasd** utility shows status information to standard output and error information to standard error in verbose mode.

Standard output

When the **-v** flag is specified, the status information of this command is written to the standard output.

Standard error

When the **-v** flag is specified, the error information of this command is written to the 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 are required to temporarily grant the clients of the cluster security service the ability to start the **ctcas** subsystem if it is not already active on the system. Without this permission and ownership, some clients of cluster security services might 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 and **rsct.core** Linux package.

Location

/usr/sbin/rsct/bin/ctstrtcasd

ctsvhbac Command

Purpose

Verifies the configuration for the RSCT host-based authentication (HBA) security mechanism on the local system.

Syntax

ctsvhbac [[**-d** | **-h** | **-m** | **-s**] | [**-e** *msgnum*[,*msgnum*...]] [**-l** { **1** | **2** | **3** | **4** } | **-b**] [**-p** *pubkeyfile*] [**-q** *pvtkeyfile*] [**-t** *thlfile*]]

Description

The **ctsvhbac** command is a verification utility for the RSCT host-based authentication (HBA) security mechanism. Use the **ctsvhbac** command to verify that the local system has configuration and credential files and information, such as private keys and a trusted host list, ready for the HBA security mechanism to use.

This command performs the following series of tests on the configuration of the HBA security mechanism:

- Verifies that the HBA mechanism configuration file is available and can be processed.
- Verifies that the HBA private key file exists and can be processed.
- Verifies that the HBA public key file exists and can be processed.
- Verifies that the private and public keys for the local system are in pair, which means that the public key is known to be derived from the private key.
- Verifies that the HBA trusted host list file exists and can be processed.
- Checks the contents of the HBA 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.

The command user may 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 HBA security mechanism.

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 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.

-p 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 HBA security mechanism.

-q 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 HBA security mechanism.

-s Displays a summary of the purpose for the command.

-t 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 HBA security mechanism.

Parameters

None.

Security

Permissions on the **ctsvhbac** command permit members of the **bin** user group to execute this command.

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 system 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 system.
- 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 HBA 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 system.

Unless these conditions are corrected, authentication requests using the HBA 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.

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.

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.

Examples

To verify the HBA security mechanism, enter:

```
ctsvhbac
```

Output would be similar to:

```
------------------------------------------------------------------------
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.pok.ibm.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
```

Location

/usr/sbin/rsct/bin/ctsvhbac
Contains the **ctsvhbac** command

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.

ctsvhbal Command

Purpose

Displays the possible identities that the local system may use to identify itself in RSCT host-based authentication (HBA) 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 host-based authentication (HBA) security mechanism. It displays the possible identities that the local system may use to identify itself in HBA credentials.

The HBA 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 system is to service requests from remote systems, at least one network address and host name for that remote system must appear in the trusted host list on the local system. To verify that the remote system can successfully authenticate the local system, system administrators use a combination of RSCT cluster security commands:

1. On both the local and remote system, issue the **ctsvhbac** command to verify that each system has a valid HBA security mechanism configuration.
2. On the local system, issue the **ctsvhbal** command to determine the values that the HBA security mechanism will use to identify this host to a remote system.
3. On the remote system, issue the **ctsvhbar** command, specifying the local system host name or IP address, to determine the value that the remote system will use to verify HBA credentials transmitted from the local system.
4. Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two systems 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 system can authenticate requests from the local system. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local system can successfully authenticate the remote system. To complete the verification, the following additional steps are required:

- On the remote system, issue the **ctsvhbal** command to determine the values that the HBA security mechanism will use to identify that host to the local system.
- On the local system, issue the **ctsvhbar** command, specifying the remote system host name or IP address, to determine the value that the local system will use to verify HBA credentials transmitted from the remote system.
- Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two systems 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 system to the local system.

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 only the host identities found for the local system and any errors 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 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.

Parameters

None.

Security

Permissions on the **ctsvhbal** command permit members of the **bin** user group to execute this 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 system is unable to authenticate itself to any remote systems. The local system 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 system uses are unable to obtain host names of network interfaces that the local system supports. Unless this condition is corrected, authentication requests using the HBA mechanism probably will not be successful on this system. For this exit status, the system administrator should follow the problem-resolution advice listed in the command output.

127 Unexpected failure in this 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.

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.

Examples

To display the possible identities that the local system may use to identify itself in HBA credentials, enter:

```
ctsvhbal
```

Output would be similar to:

```
ctsvhbal: The Host Based Authentication (HBA) mechanism identities for
the local system are:

                  Identity:  zathras.pok.ibm.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.
Ensure 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.
```

Location

/usr/sbin/rsct/bin/ctsvhbal
Contains the **ctsvhbal** command

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

ctsvhbar Command

Purpose

Returns the host name that the RSCT host-based authentication (HBA) 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) security mechanism. Use this command when you need to determine which host name the HBA security mechanism uses to verify credentials from a remote system.

The HBA 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 system is to service requests from remote systems, at least one network address and host name for that remote system must appear in the trusted host list on the local system. To verify that the remote system can successfully authenticate the local system, system administrators use a combination of RSCT cluster security commands:

1. On both the local and remote system, issue the **ctsvhbac** command to verify that each system has a valid HBA security mechanism configuration.
2. On the local system, issue the **ctsvhbal** command to determine the values that the HBA security mechanism will use to identify this host to a remote system.
3. On the remote system, issue the **ctsvhbar** command, specifying the local system host name or IP address, to determine the value that the remote system will use to verify HBA credentials transmitted from the local system.
4. Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two systems 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 system can authenticate requests from the local system. Because RSCT subsystems often use mutual authentication, system administrators also should verify that the local system can successfully authenticate the remote system. To complete the verification, the following additional steps are required:

- On the remote system, issue the **ctsvhbal** command to determine the values that the HBA security mechanism will use to identify that host to the local system.
- On the local system, issue the **ctsvhbar** command, specifying the remote system host name or IP address, to determine the value that the local system will use to verify HBA credentials transmitted from the remote system.
- Compare the **ctsvhbal** and **ctsvhbar** command output to determine whether the two systems 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 system to the local system.

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 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.

Parameters

hostname
The host name of a remote system.

address The network address of a remote system.

Security

Permissions on the **ctsvhbar** command permit members of the **bin** user group to execute this 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 system permits that name or address to be resolved.

127 Unexpected failure in this 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.

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.

Examples

To return the host name that the HBA 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.ibm.com
```

To return the host name that the HBA 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.pok.ibm.com
```

To return the host name that the HBA 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.ibm.com
Host name or network address: 9.127.100.101
Fully qualified host name
     used for authentication: epsilon3.ibm.com
```

Location

/usr/sbin/rsct/bin/ctsvhbar
Contains the **ctsvhbar** command

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

Least-privilege (LP) commands

The RSCT least-privilege (LP) resource manager is a client-server application that allows you to enhance the security, performance, and control of applications that require root authority to run. It controls access to root commands or scripts and controls the local or remote execution of those commands or scripts. It also provides a resource class and a command-line interface that let you define, manage, and monitor root commands and scripts as LP resources.

lpacl Information

Purpose

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

Description

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

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

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

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

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

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

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

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

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

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

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

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

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

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

runlpcmd
: Runs an LP resource.

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

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

chlpclacl
Changes the Class ACL

chlpracl
Changes the Resource ACL

chlpriacl
Changes the Resource Initial ACL

chlprsacl
Changes the Resource Shared ACL

lslpclacl
Lists the Class ACL

lslpracl
Lists the Resource ACL

lslpriacl
Lists the Resource Initial ACL

lslprsacl
Lists the Resource Shared ACL

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

Security

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

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

Implementation specifics

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

Location

/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 on. In a management domain, it is most likely that the LP resources are defined on the management server and the LP commands themselves are targeted to the managed nodes. In these examples, the Resource Shared ACL is not used

because separate permissions are required for the individual LP resources. These examples assume that the LP resources are not yet defined by using the **mklpcmd** command.

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

   ```
   chlpclacl lpadmin@LOCALHOST rwa

   chlpriacl lpadmin@LOCALHOST rwa
   ```

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

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

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

   ```
   chlpracl SysCmd1 joe@LOCALHOST r
   ```
4. There are several nodes in a peer domain. There is an LP resource called **SysCmdB1** on **nodeB** for which **joe** needs run permission. In addition, **joe** needs to have run permission from nodes **nodeA**, **nodeB**, and **nodeD**. If you run the **chlpracl** command on **nodeB**, you can use **joe@LOCALHOST** for **nodeB**, but you need to determine the node IDs for **nodeA** and **nodeD**. To obtain the node IDs, enter:

   ```
   lsrpnode -i
   ```

   The following output is displayed:

   ```
   Name    OpState RSCTVersion NodeNum NodeID

   nodeA   Online  3.1.0.0     2       48ce221932ae0062

   nodeB   Online  3.1.0.0     1       7283cb8de374d123

   nodeC   Online  3.1.0.0     4       b3eda8374bc839de

   nodeD   Online  3.1.0.0     5       374bdcbe384ed38a

   nodeE   Online  3.1.0.0     2       ba74503cea374110

   nodeF   Online  3.1.0.0     1       4859dfbd44023e13

   nodeG   Online  3.1.0.0     4       68463748bcc7e773
   ```

   Then, to give **joe** the permissions as stated earlier, run on nodeB:

   ```
   chlpracl SysCmd1 -l joe@LOCALHOST joe@0x48ce221932ae0062 \

   joe@0x374bdcbe384ed38a x
   ```

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

The types of ACLs' control access to the **IBM.LPCommands** class and its resources are 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.

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.

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 information about mapped user identifiers, see the *Administering RSCT* 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.

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.

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.

LP resource manager commands

chlpcmd Command

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.

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.

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.

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

-T Writes the command's trace messages to standard error.

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

Parameters

resource_name
Specifies the name of the LP resource to change.

attr1=*value1* **[***attr2*=*value2***...]**
Specifies one or more read/write attributes and their new values.

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 the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT**

is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_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.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
   ```

Location

/usr/sbin/rsct/bin/chlpcmd
Contains the **chlpcmd** command

lphistory Command
Purpose

Displays or clears the history list of least-privilege (LP) commands that have been run during the current resource monitoring and control (RMC) session.

Syntax

- To list a particular number of previously-issued commands:
  - On the local node:

    lphistory [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]
  - On all nodes in a domain:

    lphistory -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]
  - On a subset of nodes in a domain:

    lphistory -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-L a** | **c** | **e** | **m** | **n** | **t** | **u** | **x**] [**-h**] [**-TV**] [*num_records*]
- To clear the history list:
  - On the local node:

    lphistory -c [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]
  - On all nodes in a domain:

    lphistory -c -a [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]
  - On a subset of nodes in a domain:

    lphistory -c -n *host1*[**,***host2*...] [**-u** *user_ID*] [**-m** *mapped_ID*] [**-C** *command_name*] [**-S** *command_path*] [**-B** *MMddhhmmyyyy*] [**-E** *MMddhhmmyyyy*] [**-h**] [**-TV**]

Description

The **lphistory** command lists the history of LP commands that have been run by the least-privilege resource manager. The command history is maintained as records in the RSCT audit log. By default, only the command string (the path name plus arguments) from each audit log record is listed. The **-L** flag controls the output format of **lphistory**; use it to display specific fields as needed. The selection flags (**-B**, **-C**, **-E**, **-m**, **-S**, or **-u**) control the selection string that is passed to **lsaudrec**.

The **lphistory** command takes one optional parameter: the number of records to list. The default value of *num_records* is 10. If none of the selection flags is used, the latest number of records in the audit log (specified by *num_records*) are listed. Otherwise, the latest number of records (specified by *num_records*) from those selected by one or more of the selection flags are listed. This selection process applies to the audit records on each node specified by the **-a** flag or the **-n** flag. If neither **-a** nor **-n** is specified, the selection process applies to the audit records on the local node.

The **-B** and **-E** flags take time stamps as arguments. Time stamps are in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day of the month (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year.

You can use the wild card character (%) with identity-related arguments (*user_ID*, *mapped_ID*) and command names. The % can be placed at the beginning or end of the string, or anywhere within it. You cannot use any wild card characters when specifying *command_path*.

You can remove audit log records using the **-c** flag. If none of the selection flags is specified, all audit log records for the least-privilege resource manager are removed. Otherwise, the records selected by one or more of the selection flags are removed. The **-c** flag cannot be used with the **-L** flag or the *num_records* parameter.

Flags

-a Displays previously-issued LP commands for all nodes in the domain.

The **CT_MANAGEMENT_SCOPE** environment variable determines the scope of the cluster. If **CT_MANAGEMENT_SCOPE** is not set, management domain scope is chosen first (if a management domain exists), peer domain scope is chosen next (if a peer domain exists), and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds. For example, if a management domain and a peer domain both exist and **CT_MANAGEMENT_SCOPE** is not set, this command applies to the management domain. If you want this command to apply to the peer domain, set **CT_MANAGEMENT_SCOPE** to 2.

You cannot specify this flag with the **-n** flag.

-B *MMddhhmmyyyy*
Specifies a beginning time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), *mm* is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or after this time.

–c Clears the history of LP commands. You cannot specify this flag with the *number_of_commands* parameter or the **-n** flag.

-C *command_name*
Specifies a command name. **lphistory -C** lists or removes only those records that contain *command_name*, which is the name of a command without a fully-qualified path (**mkrsrc**, for example). You can use wild card characters in *command_name*.

-E *MMddhhmmyyyy*
Specifies an ending time stamp in the form *MMddhhmmyyyy*, where *MM* is the two-digit month (01-12), *dd* is the two-digit day (01-31), *hh* is the two-digit hour (00-23), mm is the two-digit minute (00-59), and *yyyy* is the four-digit year. The time can be truncated from right to left, except for *MM*. If not all digits are specified, the year defaults to the current year, minutes to 0, hour to 0, and day to 01. At a minimum, the month must be specified. The command lists or removes only those records that were created at or before this time.

-L a | c | e | m | n | t | u | x
By default, only the command string (path name plus arguments) from each audit log record is listed. If this flag is specified, the argument is one or more of the following letters; the fields are displayed in the same order as the letters in the flag argument.

a Displays all fields from the audit log in the following order: **t**, **u**, **m**, **n**, **x**, **c** (specifying **-L** a is the same as specifying **-L tumnxc**)

c Displays the command string (the default)

e Displays the standard error output

m Displays the mapped identity

n Displays the name of the node where the command ran

t Displays the time field

u Displays the authenticated user identity

x Displays the LP command exit status

You cannot specify this flag with the **-c** flag.

-m *mapped_ID*
Specifies a mapped identity. **lphistory -m** lists or removes only those records that contain *mapped_ID*. You can use wild card characters in *mapped_ID*.

-n *host1*[**,***host2*,**...**]
Specifies one or more nodes in the cluster on which the LP command history list is to be retrieved or cleared. (By default, the history list for the local node is retrieved or cleared.)

This flag is valid only in a management domain or a peer domain. If the **CT_MANAGEMENT_SCOPE** environment variable is not set, management domain scope is chosen first (if a management domain exists) and then peer domain scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope it finds.

You cannot specify this flag with the **-a** flag.

-S *command_path*
Specifies a command path name. **lphistory -S** lists or removes only those records that contain *command_path*, which is identical to the value of the **CommandPath** in the LPCommands class (**/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.

Parameters

num_records
Specifies the number of commands to be displayed from the history list. You can list a minimum of one command and a maximum of 100 commands. The default value is 10. You cannot specify this parameter with the **-c** flag.

Security

To run the **lphistory** command, you need write permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To list 20 LP commands that were previously issued on the local node, enter:
   ```
   lphistory 20
   ```
2. Suppose **nodeA** is in a management domain and **CT_MANAGEMENT_SCOPE** is set to **3**. To list the LP command history on **nodeA**, enter:
   ```
   lphistory -c -n nodeA
   ```
3. To display the last 15 LP commands invoked with time, user ID, mapped ID, mechanism, return code, standard error, command name, and command string, enter:
   ```
   lphistory -L a 15
   ```
4. To display the LP command names that end with `rsrc`, enter:
   ```
   lphistory -C %rsrc
   ```
5. To display the LP commands that were invoked after 11:30 PM on April 18, 2006, enter:
   ```
   lphistory -B 041823302006
   ```

Location

/usr/sbin/rsct/bin/lphistory
Contains the **lphistory** command.

lslpcmd Command

Purpose

Lists information about the least-privilege (LP) resources on one or more nodes in a domain.

Syntax

To display LP resource information:

- On the local node:

lslpcmd [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] **[-h]** **[-TV]**

- On all nodes in a domain:

  lslpcmd -a [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] **[-h]** **[-TV]**

- On a subset of nodes in a domain:

  lslpcmd -n *host1* **[**,*host2*,...] [**–A** | *resource_name1* [**,** *resource_name2* **,** ...] | **–R** *RunCmdName1* [**,** *RunCmdName2* **,** ...]] **[-h]** **[-TV]**

Description

The **lslpcmd** command displays information about LP resources on one or more nodes in a domain. LP resources are **root** commands or scripts to which users are granted access based on permissions in the LP access control lists (ACLs). Use this command to display the attributes of one or more LP commands by specifying the *resource_name1***,**[*resource_name2*,...] parameter. If you omit this parameter, the **lslpcmd** command lists the names of all of the LP commands. Use the **–A** flag to list all of the LP commands and all of their attributes and values. Use the **–R** flag to list one or more LP resources that have a particular **RunCmdName** value.

The **lslpcmd** command lists the following information about defined LP resources:

Field	Description
Name	The name of the LP resource.
CommandPath	The fully-qualified path of the LP resource.
Description	A description of the LP resource.
Lock	The lock setting. Valid values are: **0** (the lock is not set) and **1** (the lock is set).
CheckSum	The **CheckSum** value of the LP resource to which **CommandPath** points. The LP resource manager assigns a value of **0** if the LP resource does not exist or if the user did not update the **CheckSum** value after the LP resource was made available.
RunCmdName	The LP resource name that is used as a parameter with the **runlpcmd** command.
FilterScript	The path to the filter script.
FilterArg	The list of arguments to pass to **FilterScript**.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays information about one or more LP resources on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable's setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpcmd** command runs once for the first valid scope that the LP resource manager finds. For example, suppose a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpcmd –a** runs in the management domain. To run **lslpcmd –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-n *host1***[,***host2***,...]**
Specifies the node or nodes in the domain on which the LP resource is to be listed. By default,

the LP resource is changed on the local node. The **–n** flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpcmd** command runs once for the first valid scope that the LP resource manager finds.

–A Displays all of the LP resources with their attributes and values.

–R Display all attributes of the LP resources that have the same **RunCmdName** value.

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

-T Writes the command's trace messages to standard error.

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

Parameters

resource_name1[**,***resource_name2*,**...**]
Specifies one or more LP resources for which you want to display information.

Security

To run the **lslpcmd** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To list the names of all LP resources on the local node, enter:

   ```
   lslpcmd
   ```

   The output will look like this:

   ```
   lpcommand1
   lpcommand2
   ```

2. To list the names and attributes of all LP resources on the local node, enter:

   ```
   lslpcmd -A
   ```

   The output will look like this:

   ```
   Name=lpcommand1
   CommandPath=/tmp/my_command
   Description=
   Lock=1
   CheckSum=112
   RunCmdName=lpcommand1
   FilterScript=
   FilterArg=
   ---------------------------------
   Name=lpcommand2
   CommandPath=/tmp/cmds/this_command
   Description=
   Lock=0
   CheckSum=0
   RunCmdName=lpcommand2
   FilterScript=
   FilterArg=
   ---------------------------------
   ```

3. To list the attributes of the LP resource **lpcommand1** on the local node, enter:

   ```
   lslpcmd lpcommand1
   ```

   The output will look like this:

   ```
   Name=lpcommand1
   CommandPath=/tmp/my_command
   Description=
   Lock=1
   ```

```
CheckSum=100
RunCmdName=lpcommand1
FilterScript=
FilterArg=
```

4. To list the attributes of LP resources that have a **RunCmdName** value of **rpower** on the local node, enter:

```
lslpcmd -R rpower
```

The output will look like this:

```
Name=lpcommand1
CommandPath=/opt/csm/bin/rpower
Description=
Lock=1
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
FilterArg=node1,node2,node3
------------------------------
Name=lpcommand2
CommandPath=/opt/csm/bin/rpower
Description=
Lock=0
CheckSum=112
RunCmdName=rpower
FilterScript=/tmp/test1
FilterArg=node4,node5,node6
------------------------------
⋮
```

Location

/usr/sbin/rsct/bin/lslpcmd
Contains the **lslpcmd** command

mklpcmd Command

Purpose

Defines a new least-privilege (LP) resource to the resource monitoring and control (RMC) subsystem and specifies user permissions.

Syntax

mklpcmd [**-n** *host*] [**-l**] [**-c 0** | **1** | **2** | **3**] [**–R** *RunCmdName*] [**-s** *FilterScript*] [**-A** *FilterArg*] [**-h**] [**-TV**] *resource_name command_path* [*ID perm*] ...

Description

The **mklpcmd** command defines a new LP resource to the resource monitoring and control (RMC) subsystem subsystem. An LP resource is a **root** command or script to which users are granted access based on permissions in the LP access control lists (ACLs). Specify the LP resource using the *resource_name* parameter. The *command_path* parameter specifies the command or script that could be run with LP access. Specify the complete path name of the command or the script. If *command_path* exists when a resource is created, the LP resource manager calculates the **CheckSum** and assigns the **CheckSum** attribute value. If *command_path* does not exist, the LP resource manager assigns **0** as the **CheckSum** attribute value.

Use the **-l** flag to lock the LP resource. The resource must be unlocked before it can be deleted. Use the **-c** flag to specify the control settings of the resource.

You can also use the **mklpcmd** command to specify permissions for users when you are creating a resource. To do this, you need to have administrator permission on the resources. Administrator permission gives you the ability to set and edit permissions. You can specify multiple user IDs and permissions with this command. See the **Examples** section for more information.

This command runs on any node. In a management domain or a peer domain, use the **-n** flag to define the LP resource on the node that is specified by *host*. Otherwise, this command runs on the local node.

Flags

-n *host* Specifies the node in the domain on which the LP resource is to be defined. By default, the LP resource is defined on the local node. The **–n** flag is valid only in a management or peer domain. If the CT_MANAGEMENT_SCOPE variable is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **mklpcmd** command runs once for the first valid scope that the LP resource manager finds.

-l Defines the new LP resource as locked so that it cannot be changed accidentally. The resource cannot be removed from the RMC subsystem until the **Lock** attribute is unset.

If you do not specify this flag, the new resource is not locked. This is the default.

-c 0 | 1 | 2 | 3
Sets the **ControlFlags** attribute, which is used to specify the control features for an LP command. If **ControlFlags** is not specified, it is set to **1** by default. Use this flag to specify one of these values:

0 Does not validate the **CheckSum** value.

1 Does not validate the **CheckSum** value. This is the default.

2 Validates the **CheckSum** value.

3 Validates the **CheckSum** value.

When an attempt is made to run the LP resource using the **runlpcmd** command, the value of the **ControlFlags** attribute determines which checks are performed before running the command represented by the resource.

In this release of RSCT, the **ControlFlags** attribute value specifies whether the **CheckSum** value is to be validated.

In previous releases of RSCT, the **ControlFlags** attribute value also specified whether the presence of certain characters in the input arguments to **runlpcmd** were to be disallowed. Checking for these characters is no longer necessary.

To maintain compatibility with LP resources that were defined in previous releases of RSCT, the **ControlFlags** attribute values, with respect to validating the **CheckSum** value, have remained the same. Consequently, values **0** and **1** indicate that the **CheckSum** value is not to be validated, and values **2** and **3** indicate that the **CheckSum** value is to be validated.

-R *RunCmdName*
Specifies the **RunCmdName** value for this resource, which will be used as a parameter of the **runlpcmd** command.

-s *script_path*
Specifies the fully-qualified path of the filter script.

-A *argument*
Specifies a string of arguments to be passed to the filter script.

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

-T Writes the command's trace messages to standard error.

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

Parameters

resource_name
Is the name or identifier of the LP resource that is to be defined to the RMC subsystem.

command_path
Is the complete, fully-qualified path name of the command or script.

ID perm **...**
Specifies permissions for users when you are creating a resource. This parameter is optional.

ID Specifies the user identity for the ACL entry. See the **User identities** section of the **lpacl** information for the valid forms of this parameter.

perm Specifies the user permissions for the ACL entry. This parameter can consist of a combination of any of the following values:

r Read permission (consists of the **q**, **l**, **e**, and **v** permissions)

w Write permission (consists of the **d**, **c**, **s**, and **o** permissions)

a Administrator permission

x Execute permission

q Query permission

l Enumerate permission

e Event permission

v Validate permission

d Define and undefine permission

c Refresh permission

s Set permission

o Online, offline, and reset permission

0 No permission

See the **User permissions** section of the **lpacl** information for descriptions of these permissions.

Security

- To run the **mklpcmd** command with one or more *ID*:*perm* parameters, you need:
  - read and write permission in the Class ACL of the **IBM.LPCommands** resource class.
  - read and administrator permission in the Resource Initial ACL.

    As an alternative, the Resource Initial ACL can direct the use of the Resource Shared ACL if these permissions exist in the Resource Shared ACL.
- To run the **mklpcmd** command with no *ID*:*perm* parameters, you need write permission in the Class ACL of the **IBM.LPCommands** resource class.

Permissions are specified in the LP ACLs on the contacted system. See the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT

Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon to process the LP resource. The management scope determines the set of possible target nodes where the resource can be processed. The valid values are:

0 Specifies *local* scope.
1 Specifies *local* scope.
2 Specifies *peer domain* scope.
3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To create an LP resource called **LP1** that points to a command called **/tmp/user1/lpcmd1** on the local node, enter:

   ```
   mklpcmd LP1 /tmp/user1/lpcmd1
   ```

2. To create an LP resource called **LP2** that points to a command called **/tmp/my_command1** on **nodeB** in the management domain, enter:

   ```
   mklpcmd -n nodeB LP2 /tmp/my_command1
   ```

3. To create an LP resource called **lp3** with **ControlFlags** set to **3** (which means verify the **CheckSum** value), enter:

   ```
   mklpcmd -c 3 LP3 /tmp/cmd_lp3
   ```

4. To create an LP resource called **lp4** that points to **/tmp/testscript**, has a **RunCmdName** value of **test**, a **FilterScript** value of **/tmp/filterscr**, and filter arguments **node1** and **node2**, enter:

   ```
   mklpcmd -R test -f /tmp/filterscr -A "node1,node2" lp4 /tmp/testscript
   ```

5. To create an LP resource called **lp5** that points to **/usr/bin/mkrsrc** and gives users **user1@LOCALHOST** and **user2@LOCALHOST** read, write, and execute permission, enter:

   ```
   mklpcmd lp5 /usr/bin/mkrsrc user1@LOCALHOST rwx  user2@LOCALHOST rwx
   ```

Location

/usr/sbin/rsct/bin/mklpcmd
Contains the **mklpcmd** command

rmlpcmd Command
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.

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.

Parameters

resource_name1[**,***resource_name2*,**...**]
Specifies one or more LP resources to be removed.

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 the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resource. The management scope determines the set of possible target nodes where the resource can be processed. The valid values are:
0 Specifies *local* scope.
1 Specifies *local* scope.
2 Specifies *peer domain* scope.
3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To remove an LP resource named **LP1**, enter:
   ```
   rmlpcmd LP1
   ```
2. To remove LP resources **LP1** and **LP2**, enter:
   ```
   rmlpcmd LP1 LP2
   ```

Location

/usr/sbin/rsct/bin/rmlpcmd
Contains the **rmlpcmd** command

runlpcmd Command

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.

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.

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.

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 the **lpacl** file for general information about LP ACLs and the *RSCT Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system that is used for the session with the RMC daemon. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If 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_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

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"
```

Location

/usr/sbin/rsct/bin/runlpcmd
Contains the **runlpcmd** command

LP access control list (ACL) commands

chlpclacl Command
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.

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.

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 the **lpacl** information file.

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 the **lpacl** information file for descriptions of these permissions.

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 the **lpacl** information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system 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.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
   ```

Location

/usr/sbin/rsct/bin/chlpclacl
Contains the **chlpclacl** command

chlpracl Command
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 the **lpacl** information file.

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.

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.

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 the **lpacl** information file.

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 **lpacl** information file for a description of each permission and how it applies.

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 the **lpacl** information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT

Determines the system 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.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
    ```

Location

/usr/sbin/rsct/bin/chlpracl
Contains the **chlpracl** command

chlpriacl Command

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.

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.

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 **lpacl** information file.

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 **lpacl** information file for a description of each permission and how it applies.

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 the **lpacl** information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system 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.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
   ```

Location

/usr/sbin/rsct/bin/chlpriacl
Contains the **chlpriacl** command

chlprsacl Command

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.

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.

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 **lpacl** information file.

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 **lpacl** information file for a description of each permission and how it applies.

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 the **lpacl** information file for general information about LP ACLs and the *RSCT: Administration Guide* for information about modifying them.

Exit Status

0 The command has run successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Environment Variables

CT_CONTACT
Determines the system 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.

Implementation Specifics

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

Standard Output

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

Standard Error

All trace messages are written to standard error.

Examples

1. To 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
   ```

Location

/usr/sbin/rsct/bin/chlprsacl
Contains the **chlprsacl** command

lslpclacl Command
Purpose

Displays the access controls for the least-privilege (LP) resource class (**IBM.LPCommands**).

Syntax

To display the access controls for the **IBM.LPCommands** resource class:

- On the local node:
  lslpclacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On all nodes in a domain:
  lslpclacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On a subset of nodes in a domain:
  lslpclacl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

Description

The **lslpclacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) resource class (**IBM.LPCommands**). The accesses contained in the ACL entries are displayed. The **IBM.LPCommands** Class ACL controls access to the **IBM.LPCommands** class operations. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description	
Identity	The network identity of the user. See the lpacl command for a description of the network identity.	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays the **IBM.LPCommands** Class ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpclacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpclacl –a** runs in the management domain. To run **lslpclacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpclacl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). It is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (|) when the information that you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[**,***host2*,**...**]
Specifies the node in the domain from which the **IBM.LPCommands** Class ACL is displayed. By default, the **IBM.LPCommands** Class ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based

network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the **-a** flag or the **-n** flag is specified.

Standard output

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

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpclacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 279 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

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

Location

/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 following output is displayed:
   ```
   Identity                        Permissions     NodeName

   joe@LOCALHOST                   ra              nodeA

   bill@0x374bdcbe384ed38a         rwa             nodeA

   jane@0x374bdcbe384ed38a         rwa             nodeA
   ```
2. To list the **IBM.LPCommands** Class ACLs on **nodeA** in long format, run this command on **nodeA**:
   ```
   lslpclacl -l
   ```
   The following output is displayed:
   ```
   Class ACLs for LPRM

   NodeName nodeA

       Identity    =     joe@LOCALHOST

       Permissions =     ra


       Identity    =     bill@0x374bdcbe384ed38a

       Permissions =     rwa


       Identity    =     jane@0x374bdcbe384ed38a

       Permissions =     rwa
   ```
3. To list the **IBM.LPCommands** Class ACLs on **nodeA** in delimited format, run this command on **nodeA**:
   ```
   lslpclacl -d
   ```
   The following output is displayed:
   ```
   Identity|Permissions|NodeName

   joe@LOCALHOST|ra|nodeA

   bill@0x374bdcbe384ed38a|rwa|nodeA

   jane@0x374bdcbe384ed38a|rwa|nodeA
   ```
4. To list the **IBM.LPCommands** Class ACLs on **nodeA** in the active domain, run this command:
   ```
   lslpclacl -a
   ```
   The following output is displayed:
   ```
   Identity                        Permissions     NodeName

   joe@LOCALHOST                   ra              node1.pok.ibm.com

   bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com

   jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com
   ```

```
joe@LOCALHOST                   ra              node2.pok.ibm.com

jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com
```

5. To list the **IBM.LPCommands** Class ACLs on **nodeA** in the active domain and list the peer domain name, run this command:

```
lslpclacl -ap
```

The following output is displayed:

```
Identity                        Permissions     NodeName                PeerDomain

joe@LOCALHOST                   ra              node1.pok.ibm.com       PD1

bill@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwa             node1.pok.ibm.com       PD1

joe@LOCALHOST                   ra              node2.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwa             node2.pok.ibm.com       PD1
```

lslpracl Command

Purpose

Displays the access controls for a least-privilege (LP) resource.

Syntax

To display the access controls for an LP resource:

- On the local node:
  lslpracl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-L**] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**] [*name*]
- On all nodes in a domain:
  lslpracl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-L**] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**] [*name*]
- On a subset of nodes in a domain:
  lslpracl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-L**] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**] [*name*]

Description

The **lslpracl** command displays the access control list (ACL) that is associated with a least-privilege (LP) resource. The accesses contained in the ACL entries are displayed. The Resource ACL controls access to the LP resources. If no LP resource name is specified, the Resource ACLs for all LP resources are listed. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description
Name	The name of the LP resource. See "lpacl Information" on page 279 for a description of the network identity.
Identity	The network identity of the user.

Field	Description	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the LP resource (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the LP resource is defined. This field is displayed when the **-p** flag is specified.	

If the Resource ACL indicates that the Resource Shared ACL controls access to the LP resource, the ID is displayed as **Uses Resource Shared ACL** and there is no permission value. Use the **-L** flag to display the Resource Shared ACL when it is used by the Resource ACLs that are being displayed.

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Parameters

name Specifies the name of the LP resource.

Flags

-a Displays the Resource ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpracl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpracl –a** runs in the management domain. To run **lslpracl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpracl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). It is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[**,***host2*,**...**]
Specifies the node in the domain from which the Resource ACL is displayed. By default, the Resource ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-L Displays the accesses of the Resource Shared ACL if the Resource ACL indicates that access is controlled by the Resource Shared ACL.

-p Displays the name of the RSCT peer domain in which the LP resource is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the **-a** flag or the **-n** flag is specified.

Standard output

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

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpracl** command, you need:

- read permission in the Class ACL of the **IBM.LPCommands** resource class.
- read permission in the Resource ACL.

  As an alternative, the Resource ACL can direct the use of the Resource Shared ACL if this permission exists in the Resource Shared ACL.

Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 279 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

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

Location

/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 following output is displayed:

   ```
   Resource ACLs for LPRM

   Name            Identity               Permissions     NodeName

   lpcommand1      joe@LOCALHOST          rx              nodeA
   ```

```
lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA

lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA
```

2. To list the Resource ACLs for the LP resource **lpcommand1** on **nodeA** in long format, run this command on **nodeA**:

```
lslpracl -l lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM

Name lpcommand1, NodeName nodeA

    Identity    =     joe@LOCALHOST

    Permissions =     rx


    Identity    =     bill@0x374bdcbe384ed38a

    Permissions =     rx


    Identity    =     jane@0x374bdcbe384ed38a

    Permissions =     rwax
```

3. To list the Resource ACLs for the LP resource **lpcommand1** on **nodeA** in delimited format, run this command on **nodeA**:

```
lslpracl -d lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM

Name|Identity|Permissions|NodeName

lpcommand1|joe@LOCALHOST|rx|nodeA

lpcommand1|bill@0x374bdcbe384ed38a|rx|nodeA

lpcommand1|jane@0x374bdcbe384ed38a|rwax|nodeA
```

4. To list the Resource ACLs for the LP resource **lpcommand1** in the active domain, run this command on **nodeA**:

```
lslpracl -a lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permissions     NodeName

lpcommand1      joe@LOCALHOST           rx              nodeA.pok.ibm.com

lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA.pok.ibm.com

lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA.pok.ibm.com

lpcommand1      joe@LOCALHOST           rx              nodeB.pok.ibm.com

lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeB.pok.ibm.com
```

5. To list the Resource ACLs for all LP resources on **nodeA**, run this command on **nodeA**:

```
lslpracl
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permissions     NodeName

lpcommand1      joe@LOCALHOST           rx              nodeA

lpcommand1      bill@0x374bdcbe384ed38a rx              nodeA

lpcommand1      jane@0x374bdcbe384ed38a rwax            nodeA

lpcommand2      jim@LOCALHOST           rx              nodeA

lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA

lpcommand3      mary                    rwax            nodeA

lpcommand4      bob@LOCALHOST           rx              nodeA

lpcommand4      sam@0x374bdcbe384ed38a  rwax            nodeA
```

6. To list the Resource ACLs for the LP resource **lpcommand1** in the active domain and list the peer domain name, run this command on **nodeA**:

```
lslpracl -ap lpcommand1
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permission NodeName            PeerDomain

lpcommand1      joe@LOCALHOST           rx         nodeA.pok.ibm.com   PD1

lpcommand1      bill@0x374bdcbe384ed38a rx         nodeA.pok.ibm.com   PD1

lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeA.pok.ibm.com   PD1

lpcommand1      joe@LOCALHOST           rx         nodeB.pok.ibm.com   PD1

lpcommand1      jane@0x374bdcbe384ed38a rwax       nodeB.pok.ibm.com   PD1
```

7. To list the Resource ACLs for the LP resource **lpcommand2** on **nodeA**, run this command on **nodeA**:

```
lslpracl lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                    Permissions   NodeName

lpcommand2      Uses Resource Shared ACL                  nodeA
```

8. To list the Resource ACLs for the LP resource **lpcommand2** on **nodeA**, and show the Resource Shared ACL if it is used, run this command on **nodeA**:

```
lslpracl -L lpcommand2
```

The following output is displayed:

```
Resource ACLs for LPRM

Name            Identity                Permissions     NodeName

lpcommand2      bill@0x374bdcbe384ed38a rx              nodeA

lpcommand2      jane@0x374bdcbe384ed38a rwax            nodeA
```

lslpriacl Command
Purpose

Displays the access controls for the least-privilege (LP) Resource Initial ACL.

Syntax

To display the access controls for the Resource Initial ACL:

- On the local node:
  lslpriacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On all nodes in a domain:
  lslpriacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]
- On a subset of nodes in a domain:
  lslpriacl { **-n** *host1*[**,***host2*,...] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] [**-p**] [**-E**] [**-x**] [**-h**] [**-TV**]

Description

The **lslpriacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Initial ACL. The accesses contained in the ACL entries are displayed. The Resource Initial ACL is used as the Initial ACL that gets copied to the Resource ACL when an LP resource is created. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description	
Identity	The network identity of the user. See "lpacl Information" on page 279 for a description of the network identity.	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays the Resource Initial ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslpriacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslpriacl –a** runs in the management domain. To run **lslpriacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlpriacl** command.

-l Displays the information about separate lines (long format).

-t Displays the information in separate columns (table format). This is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other than the default pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1***[,***host2***,...]**
Specifies the node in the domain from which the Resource Initial ACL is displayed. By default, the Resource Initial ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based

network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the **-a** flag or the **-n** flag is specified.

Standard output

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

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslpriacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 279 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

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

Location

/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 following output is displayed:
   ```
   Resource Initial ACLs for LPRM

   Identity                        Permissions     NodeName

   joe@LOCALHOST                   rx              nodeA

   bill@0x374bdcbe384ed38a         rwx             nodeA

   jane@0x374bdcbe384ed38a         rwax            nodeA
   ```
2. To list the Resource Initial ACLs on **nodeA** in long format, run this command on **nodeA**:
   ```
   lslpriacl -l
   ```

   The following output is displayed:
   ```
   Resource Initial ACLs for LPRM

   NodeName c175n06.ppd.pok.ibm.com

      Identity    =     joe@LOCALHOST

      Permissions =     rx


      Identity    =     bill@0x374bdcbe384ed38a

      Permission  =     rwx


      Identity    =     jane@0x374bdcbe384ed38a

      Permissions =     rwax
   ```
3. To list the Resource Initial ACLs on **nodeA** in delimited format, run this command on **nodeA**:
   ```
   lslpriacl -d
   ```

   The following output is displayed:
   ```
   Resource Initial ACLs for LPRM

   Identity|Permissions|NodeName

   joe@LOCALHOST|rx|nodeA

   bill@0x374bdcbe384ed38a|rwx|nodeA

   jane@0x374bdcbe384ed38a|rwax|nodeA
   ```
4. To list the Resource Initial ACLs in the active domain, run this command:
   ```
   lslpriacl -a
   ```

   The following output is displayed:
   ```
   Resource Initial ACLs for LPRM

   Identity                        Permissions     NodeName

   joe@LOCALHOST                   rx              nodeA.pok.ibm.com
   ```

```
bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com

jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com

joe@LOCALHOST                   rx              nodeB.pok.ibm.com

jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com
```

5. To list the Resource Initial ACLs in the active domain and list the peer domain name, run this command:

```
lslpriacl -ap
```

The following output is displayed:

```
Resource Initial ACLs for LPRM

Identity                        Permissions     NodeName                PeerDomain

joe@LOCALHOST                   rx              nodeA.pok.ibm.com       PD1

bill@0x374bdcbe384ed38a         rwx             nodeA.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwax            nodeA.pok.ibm.com       PD1

joe@LOCALHOST                   rx              nodeB.pok.ibm.com       PD1

jane@0x374bdcbe384ed38a         rwax            nodeB.pok.ibm.com       PD1
```

lslprsacl Command

Purpose

Displays the access controls for the least-privilege (LP) Resource Shared ACL.

Syntax

To display the access controls for the Resource Shared ACL:

- On the local node:
  lslprsacl [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]**
- On all nodes in a domain:
  lslprsacl -a [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]**
- On a subset of nodes in a domain:
  lslprsacl { **-n** *host1*[*,host2,...*] } [**-l** | **-i** | **-t** | **-d** | **-D** *delimiter*] **[-p]** **[-E]** **[-x]** **[-h]** **[-TV]**

Description

The **lslprsacl** command displays the access control list (ACL) that is associated with the least-privilege (LP) Resource Shared ACL. The accesses contained in the ACL entries are displayed. The Resource Shared ACL controls access to LP resources in which the Resource ACL indicates that the Resource Shared ACL is used. By default, this command displays information in table format (**-t**).

This command displays the following ACL information:

Field	Description	
Identity	The network identity of the user. See "lpacl Information" on page 279for a description of the network identity.	
Permissions	The permissions allowed for **Identity**. The valid values are:	
	a	Administrator permission
	r	Read permission (consists of the **e**, **l**, **q**, and **v** permissions)
	w	Write permission (consists of the **c**, **d**, **o**, and **s** permissions)
	x	Execute permission
	c	Refresh permission
	d	Define and undefine permission
	e	Event permission
	l	Enumerate permission
	o	Online, offline, and reset permission
	q	Query permission
	s	Set permission
	v	Validate permission
	0	No permission
NodeName	The location of the **IBM.LPCommands** resource class (for management domain scope or peer domain scope).	
PeerDomain	The name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined. This field is displayed when the **-p** flag is specified.	

This command runs on any node. If you want this command to run on all of the nodes in a domain, use the **-a** flag. If you want this command to run on a subset of nodes in a domain, use the **-n** flag. Otherwise, this command runs on the local node.

Flags

-a Displays Resource Shared ACLs on all nodes in the domain. The **CT_MANAGEMENT_SCOPE** environment variable setting determines the cluster scope. If **CT_MANAGEMENT_SCOPE** is not set, the LP resource manager uses scope settings in this order:

1. The management domain, if it exists
2. The peer domain, if it exists
3. Local scope

The **lslprsacl** command runs once for the first valid scope that the LP resource manager finds. For example, suppose that a management domain and a peer domain exist and the **CT_MANAGEMENT_SCOPE** environment variable is not set. In this case, **lslprsacl –a** runs in the management domain. To run **lslprsacl –a** in the peer domain, you must set **CT_MANAGEMENT_SCOPE** to **2**.

-i Generates a template in a form that can be used, after appropriate editing, as file input to the **chlprsacl** command.

-l Displays the information on separate lines (long format).

-t Displays the information in separate columns (table format). This is the default.

-d Displays the information using delimiters. The default delimiter is a pipe symbol (|). Use the **-D** flag if you want to change the default delimiter.

-D *delimiter*
Displays the information using the specified delimiter. Use this flag to specify a delimiter other

than the default pipe symbol (|) when the information you want to display contains pipe symbols, for example. You can use this flag to specify a delimiter of one or more characters.

-n *host1*[**,***host2*,**...**]
Specifies the node in the domain from which the Resource Shared ACL is displayed. By default, the Resource Shared ACL is displayed on the local node. This flag is valid only in a management domain or a peer domain. If **CT_MANAGEMENT_SCOPE** is not set, first the management domain scope is chosen if it exists, then the peer domain scope is chosen if it exists, and then local scope is chosen, until the scope is valid for the command. The command runs once for the first valid scope found.

-p Displays the name of the RSCT peer domain in which the **IBM.LPCommands** resource class is defined.

-E Displays read permission as **elqv** instead of **r** and write permission as **cdos** instead of **w**.

-x Excludes the header (suppresses header printing).

-h Writes the command usage statement to standard output.

-T Writes the command trace messages to standard error.

-V Writes the command verbose messages to standard output.

Environment variables

CT_CONTACT
Determines the system where the session with the resource monitoring and control (RMC) daemon occurs. When **CT_CONTACT** is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If **CT_CONTACT** is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the resource classes or resources that are processed.

CT_IP_AUTHENT
When the CT_IP_AUTHENT environment variable exists, the RMC daemon uses IP-based network authentication to contact the RMC daemon on the system that is specified by the IP address to which the CT_CONTACT environment variable is set. CT_IP_AUTHENT has meaning only if CT_CONTACT is set to an IP address; it does not rely on the domain name system (DNS) service.

CT_MANAGEMENT_SCOPE
Determines the management scope that is used for the session with the RMC daemon in processing the resources of the least-privilege (LP) resource manager. The management scope determines the set of possible target nodes where resources can be processed. The valid values are:

0 Specifies *local* scope.

1 Specifies *local* scope.

2 Specifies *peer domain* scope.

3 Specifies *management domain* scope.

If this environment variable is not set, *local* scope is used, unless the **-a** flag or the **-n** flag is specified.

Standard output

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

Standard error

All trace messages are written to standard error.

Exit status

0 The command ran successfully.

1 An error occurred with RMC.

2 An error occurred with the command-line interface (CLI) script.

3 An incorrect flag was specified on the command line.

4 An incorrect parameter was specified on the command line.

5 An error occurred with RMC that was based on incorrect command-line input.

6 The resource was not found.

Security

To run the **lslprsacl** command, you need read permission in the Class ACL of the **IBM.LPCommands** resource class. Permissions are specified in the LP ACLs on the contacted system. See "lpacl Information" on page 279 for general information about LP ACLs and the *Administering RSCT* guide for information about modifying them.

Implementation specifics

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

Location

/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 following output is displayed:

   ```
   Resource Shared ACLs for LPRM

   Identity                        Permissions     NodeName

   joe@LOCALHOST                   rx              nodeA

   bill@0x374bdcbe384ed38a         rwx             nodeA

   jane@0x374bdcbe384ed38a         rwax            nodeA
   ```

2. To list the Resource Shared ACLs on **nodeA** in long format, run this command on **nodeA**:

   ```
   lslprsacl -l
   ```

   The following output is displayed:

   ```
   Resource Shared ACLs for LPRM

   NodeName c175n06.ppd.pok.ibm.com

      Identity    =     joe@LOCALHOST

      Permissions =     rx
   ```

```
Identity    =     bill@0x374bdcbe384ed38a

Permissions =     rwx


Identity    =     jane@0x374bdcbe384ed38a

Permissions =     rwax
```

3. To list the Resource Shared ACLs on **nodeA** in delimited format, run this command on **nodeA**:
   ```
   lslprsacl -d
   ```

   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   Identity|Permissions|NodeName

   joe@LOCALHOST|rx|nodeA

   bill@0x374bdcbe384ed38a|rwx|nodeA

   jane@0x374bdcbe384ed38a|rwax|nodeA
   ```
4. To list the Resource Shared ACLs in the active domain, run this command:
   ```
   lslprsacl -a
   ```

   The following output is displayed:
   ```
   Identity                      Permissions     NodeName

   joe@LOCALHOST                 rx              nodeA.pok.ibm.com

   bill@0x374bdcbe384ed38a       rwx             nodeA.pok.ibm.com

   jane@0x374bdcbe384ed38a       rwax            nodeA.pok.ibm.com

   joe@LOCALHOST                 rx              nodeB.pok.ibm.com

   jane@0x374bdcbe384ed38a       rwax            nodeB.pok.ibm.com
   ```
5. To list the Resource Shared ACLs in the active domain and list the peer domain name, run this command:
   ```
   lslprsacl -ap
   ```

   The following output is displayed:
   ```
   Resource Shared ACLs for LPRM

   Identity                      Permissions     NodeName              PeerDomain

   joe@LOCALHOST                 rx              nodeA.pok.ibm.com     PD1

   bill@0x374bdcbe384ed38a       rwx             nodeA.pok.ibm.com     PD1

   jane@0x374bdcbe384ed38a       rwax            nodeA.pok.ibm.com     PD1

   joe@LOCALHOST                 rx              nodeB.pok.ibm.com     PD1

   jane@0x374bdcbe384ed38a       rwax            nodeB.pok.ibm.com     PD1
   ```

Subsystem control and status commands

This section explains **cthactrl** and **nlssrc** commands that are used to control and give status of the subsystems respectively.

cthactrl Command

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.

  For example:

```
cthactrl -i -c filesys -n 1 -e filesys -p "cthats=12347,cthags=12348"
```

-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
: Uninstalls the group services and topology services subsystems.

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

Security

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

Exit Status

0
: Successful completion.

non-zero
A failure has occurred.

Restrictions

This command applies to the **cthags** and **cthats** subsystems only.

Standard Output

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

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
   ```

Location

/usr/sbin/rsct/bin/cthactrl

nlssrc Command

Purpose

Gets the status of a subsystem or a group of subsystems in canonical form.

Syntax

nlssrc [**-h** *host*] **-a**

nlssrc [**-h** *host*] **-g** *group_name*

nlssrc [**-h** *host*] [**-l**] [**-c**] **-s** *subsystem_name*

nlssrc [**-h** *host*] [**-l**] [**-c**] **-p** *subsystem_pid*

The syntax for the first two usages of **nlssrc** will generate the exact same output as **lssrc**. The syntax for the last two usages will generate the output in the canonical form as **lssrc**.

Description

Use the **nlssrc** command to get the status of a subsystem or a group of subsystems in canonical form. For the AIX platform, use the **nlssrc -c** command 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 present, the **nlssrc** command will invoke the **lssrc** command that uses the daemon's locale.

Flags

Item	Description
-a	Lists 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 to get status for. The command is unsuccessful if the *group_name* parameter is not contained in the subsystem object class.
-h *host*	Specifies the foreign host on which this status action is requested. The local user must be running as root. The remote system must be configured to accept remote System Resource Controller (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 to get status for. The *subsystem_name* parameter can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *subsystem_name* parameter is not contained in the subsystem object class.

Security

You do *not* need **root** authority to run this command.

Exit Status

0 Command has run successfully.

1 Command was not successful.

Restrictions

This command applies to the **cthags** and **cthats** subsystems only.

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).

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:

   ```
   nlssrc -ls ctsubsys (locale-dependent)

   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

nlssrc -ls ctsubsys -c (canonical form)

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
```

Location

/usr/sbin/rsct/bin/nlssrc
Contains the **nlssrc** command

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.

Topology Services commands

In an RSCT peer domain, the configuration resource manager uses the Topology Services subsystem to monitor the liveness of the adapters and networks included in communication groups.

Topology Services is a distributed subsystem that provides information to other subsystems about the state of the nodes and adapters in the cluster.

cthatsctrl Command

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.

Security

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

Exit Status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

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

Standard Error

This command writes any error messages to standard error.

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
```

Location

/usr/sbin/rsct/bin/cthatsctrl
Contains the **cthatsctrl** command.

cthatstune Command

Purpose

Views and changes the topology services subsystem's tunable parameters at run time.

Syntax

cthatstune [**-f** [*network1*]**:***frequency1*[**,**[*network2*]**:***frequency2*...]] [**-g** [[*network*]**:***grace*]] [**-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**.

-g [[*network***]:***grace***]**
Specifies the grace period that is used when heartbeats are no longer received. When a heartbeat is missed, an Internet Control Message Protocol (ICMP) echo packet is sent to the failed node. If the echo is returned, the grace period is initiated.

The grace period is specified in seconds and is significant to milliseconds. It can be specified as an integer, a floating-point number, or one of these values:

0 Specifies that the grace period is disabled.

-1 | d Specifies that the topology services subsystem controls the grace period. This is the default value.

-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.

Security

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

Exit Status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

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.

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.

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
   ```

6. To change the period for network communication group **CG3** to 2345 milliseconds, enter:

   ```
   cthatstune -f CG3:2.345
   ```

7. To change the grace period for network communication group **CG3** to 30500 milliseconds, enter:

   ```
   cthatstune -g CG3:30.5
   ```

Location

/usr/sbin/rsct/bin/cthatstune
Contains the **cthatstune** command

hatsoptions Command

Purpose

Controls topology services options on a node or a control workstation.

Syntax

hatsoptions [**-s**] [**-d**]

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.

Flags

-s Instructs the topology services daemon to reject messages that are apparently delayed.

-d Instructs the topology services daemon not to reject messages that are apparently delayed (this is the default).

Security

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

Exit Status

0 Indicates the successful completion of the command.

1 Indicates the command was unsuccessful.

Environment Variables

HB_SERVER_SOCKET
This environment variable should be set before this command can be executed. It must be set to the location of the UNIX domain socket used by topology services clients to connect to the topology services daemon. This environment variable must be set to **/var/ha/soc/hats/server_socket.***partition name*.

HA_SYSPAR_NAME
If HB_SERVER_SOCKET is not set, then HA_SYSPAR_NAME must be set to the partition name.

Restrictions

This command is valid in a peer domain only.

Standard Output

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

Standard Error

This command writes error messages (as necessary) to standard error.

Examples

To instruct the topology services daemon on the local node to start discarding apparently delayed messages, enter:

```
export HA_SYSPAR_NAME=partition1

/usr/sbin/rsct/bin/hatsoptions -s
```

Location

/usr/sbin/rsct/bin/hatsoptions
Contains the **hatsoptions** command

Files

/var/ha/soc/hats/server_socket.*partition name*

Group services commands

The configuration resource manager uses the Group Services subsystem to provide distributed coordination, messaging, and synchronization among nodes in an RSCT peer domain.

Group services provides distributed coordination and synchronization services for other distributed subsystems running on a set of nodes in a cluster.

cthagsctrl Command

Purpose

Controls the group services subsystem.

Syntax

cthagsctrl { **-a** [**-p** *port-number*] **-s** | **-k** | **-d** | **-r** | **-z** | **-h** | **-t** | **-o** }

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 subsystems
cthags

The **cthags** subsystem is 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 subsystems:

- Adding, starting, stopping, and deleting the subsystems
- Cleaning up the subsystems (deleting them from the cluster)
- Unconfiguring the subsystems from the cluster
- Turning tracing on and off

Adding the subsystem

When the **-a** flag is specified, the control command adds the group services subsystems to the SRC. The control 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.

Starting the subsystem

When the **-s** flag is specified, the control command uses the **startsrc** command to start the group services subsystem, **cthags**.

Stopping the subsystem

When the **-k** flag is specified, the control command uses the **stopsrc** command to stop the group services subsystem, **cthags**.

Deleting or cleaning the subsystem

When the **-d** flag is specified, the control 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.

Turning tracing on

When the **-t** flag is specified, the control command turns tracing on for the **hagsd** 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 **hagsd** daemon using the **tracesoff** command.

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.

-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.

-r
: Refreshes the subsystem.

-z
: Uninstalls the **cthags** subsystem.

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

Security

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

Exit Status

0
: Indicates that the command completed successfully.

a non-zero value
: Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

Use this command *only* under the direction of the IBM Support Center.

Standard Output

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

Standard Error

This command writes error messages, as necessary, to standard error.

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 start the group services subsystems in the current cluster, enter:
   ```
   cthagsctrl -s
   ```
4. To stop the group services subsystems in the current cluster, enter:
   ```
   cthagsctrl -k
   ```
5. To delete the group services subsystems from the SRC in the current cluster, enter:
   ```
   cthagsctrl -d
   ```
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
   ```

Location

/usr/sbin/rsct/bin/cthagsctrl
Contains the **cthagsctrl** command

cthagstune Command

Purpose

Changes the group services subsystem tunable parameters at run time.

Syntax

cthagstune [**-l** *log_length*] [**-d** *log_dirsize*]

cthagstune [**-h**]

Description

The **cthagstune** command changes the group services subsystem 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.

Security

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

Exit Status

0 Indicates that the command completed successfully.

a non-zero value
Indicates that an error occurred.

Restrictions

This command is valid in a peer domain only.

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.

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
```

Location

/usr/sbin/rsct/bin/cthagstune
Contains the **cthagstune** command

hagsns Command

Purpose

Gets group services name server information.

Syntax

hagsns [**-h** *host*] [**-c**] **-g** *group_name*

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.

-g *group_name*
Specifies a group of subsystems to get status for. The command is unsuccessful if the *group_name* variable is not contained in the subsystem object class.

-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.

Parameters

daemon_name
: Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Security

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

Exit Status

0 Indicates that the command completed successfully.

a non-zero value
: Indicates that an error occurred.

Restrictions

This command is valid in a PSSP environment only.

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.

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.

Location

/usr/sbin/rsct/bin/hagsns
: Contains the **hagsns** command

Files

/var/ha/log/hags_nodenum_instnum**.** *syspar_name*
Contains the log of the **hagsd** daemons on the nodes.

/var/ha/log/hags.*syspar_name_nodenum_instnum***.***syspar_name*
Contains the log of each **hagsd** daemon on the control workstation.

The file names include the following variables:

- *nodenum* is the node number on which the daemon is running.
- *instnum* is the instance number of the daemon.
- *syspar_name* is the name of the system partition in which the daemon is running.

hagsvote Command

Purpose

Gets vote information for group services groups.

Syntax

hagsvote [**-h** *host*] [**-l**] [**-a** *argument*] [**-c**] **-g** *group_name*

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 Specifies a group services group name. This group name is different from that of the **-g** flag. In this case, the group 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.

-g *group_name*
Specifies a group of subsystems to get status for. The command is unsuccessful if the *group_name* variable is not contained in the subsystem object class.

-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.

Parameters

daemon_name
Specifies the name used by the daemon to name log files and identify its messages in the AIX error log.

Security

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

Exit Status

0 Indicates the successful completion of the command.

non-zero
Indicates that an error occurred.

Restrictions

This command is valid in a PSSP environment only.

Standard Output

This command writes error messages (as necessary) to standard error.

Standard Error

This command writes error messages, as necessary, to standard error.

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 a n-phase protocol.
   - The protocol is the Join protocol.
   - For the local node, it has 1 provider, the number of providers which have not voted yet is 1.
   - 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:

   - The number of providers that have not voted yet is 1.
   - 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]
   ```

Location

/usr/sbin/rsct/bin/hagsvote
Contains the **hagsvote** command

Files

/var/ha/log/hags_*nodenum_instnum*_**.** *syspar_name*
Contains the log of the **hagsd** daemons on the nodes.

/var/ha/log/hags.*syspar_name_nodenum_instnum***.***syspar_name*
Contains the log of each **hagsd** daemon on the control workstation.

The file names include the following variables:

- *nodenum* is the node number on which the daemon is running
- *instnum* is the instance number of the daemon
- *syspar_name* is the name of the system partition in which the daemon is running.

System resource controller (SRC) commands

The system resource controller (SRC) provides a set of commands and subroutines to make it easier for the system manager and programmer to create and control subsystems.

SRC is a subsystem feature that can be used to define and control subsystems. The Group Services subsystem is called **cthags** and the Topology Services subsystem is called **cthats**. These subsystem names are used with the SRC commands (for example, **lssrc**).

lssrc Command

Purpose

Gets the status of a subsystem, a group of subsystems, or a subserver.

Syntax

To Get All Status

lssrc [**-h** *Host*] **-a**

To Get Group Status

lssrc [**-h** *Host*] **-g** *GroupName*

To Get Subsystem Status

lssrc [**-h** *Host*] [**-l**] **-s** *Subsystem*

To Get Status by PID

lssrc [**-h** *Host*] [**-l**] **-p** *SubsystemPID*

To Get Subserver Status

lssrc [**-h** *Host*] [**-l**] **-t** *Type* [**-p** *SubsystemPID*] [**-o** *Object*] [**-P** *SubserverPID*]

To Get Subsystem Status in SMIT Format

lssrc -S [**-s** *Subsystem* | **-d**]

To Get Subserver Status in SMIT Format

lssrc -T [**-t** *Type*]

To Get Notify in SMIT Format

lssrc -N [**-n** *NotifyName*]

Description

The **lssrc** command sends a request to the System Resource Controller to get status on a subsystem, a group of subsystems, or all subsystems. The **lssrc** command sends a subsystem request packet to the daemon to be forwarded to the subsystem for a subserver status or a long subsystem status.

You can choose whether to request a short or long status for a subserver. When the **-l** flag is absent, the status request is assumed to be a short status. A short status of a subsystem, group of subsystems, or all subsystems is handled by the System Resource Controller.

When the **-l** flag is present for a subsystem, a status request is taken to the subsystem and the subsystem sends the status back. The **-l** flag is supported only for those subsystems not using signals as their communication method. For either a long or short status of a subserver, the subsystem is sent a status request packet, and the subsystem sends the status back.

The **lssrc** command output can sometimes show two entries for a particular daemon. One instance will be active and another instance will be inoperative. This can happen if the subsystem is modified (using the **mkssys** command or **chssys** command) without stopping the subsystem. The original subsystem will remain active and the modified instance will be inoperative until the subsystem is stopped and started again.

Flags

Item	Description
-a	Lists the current status of all defined subsystem.
-d	Specifies that the default record is printed.
-g *GroupName*	Specifies a group of subsystems to get status for. The command is unsuccessful if the *GroupName* variable is not contained in the subsystem object class.
-h *Host*	Specifies the foreign host on which this status action is requested. The local user must be running as "root". The remote system must be configured to accept remote System Resource Controller requests. That is, the **srcmstr** daemon (see **/etc/inittab**) must be started with the **-r** flag and the **/etc/hosts.equiv** or **.rhosts** file must be configured to allow remote requests.
-l	Requests that a subsystem send its current status in long form. Long status requires that a status request be sent to the subsystem; it is the responsibility of the subsystem to return the status.
-n *NotifyName*	Specifies the name of a notify method.
-N	Specifies that the Object Data Manager (ODM) records are output in SMIT format for the notify object class.
-o *Object*	Specifies that a subserver *Object* variable is passed to the subsystem as a character string.
-p *SubsystemPID*	Specifies a particular instance of the *SubsystemPID* variable to get status for, or a particular instance of the subsystem to which the status subserver request is to be taken.
-P *SubserverPID*	Specifies that a *SubserverPID* variable is to be passed to the subsystem as a character string.
-s *Subsystem*	Specifies a subsystem to get status for. The *Subsystem* variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the *Subsystem* variable is not contained in the subsystem object class.
-S	Specifies that the ODM records are output in SMIT format for the subsystem object class.
-t *Type*	Requests that a subsystem send the current status of a subserver. The command is unsuccessful if the subserver *Type* variable is not contained in the subserver object class.
-T	Specifies that the ODM records are output in SMIT format for the subserver object class.

Security

Attention RBAC users and Trusted AIX users: This command can perform privileged operations. Only privileged users can run privileged operations. For more information about authorizations and privileges, see Privileged Command Database in *AIX Version 7.1 Security*. For a list of privileges and the authorizations associated with this command, see the **lssecattr** command or the **getcmdattr** subcommand.

Examples

1. To get the status of all subsystems on the local machine, enter:

   ```
   lssrc  -a
   ```

   This gets the status of all subsystems known on the local machine.
2. To get the status of all subsystems on a foreign host, enter:

   ```
   lssrc  -h zork  -a
   ```

   This gets the status of all subsystems known on the `zork` machine.
3. To get the status of the srctest subsystem, enter:

   ```
   lssrc  -s srctest
   ```

   This gets the status of all instances of the `srctest` subsystem on the local machine.
4. To get the status of the subsystem by PID, enter:

   ```
   lssrc  -p 1234
   ```

   This gets the status of the subsystem with the subsystem PID of `1234` on the local machine.
5. To get the status of the `tcpip` subsystem group, enter:

```
lssrc  -g tcpip
```

This gets the status of all instances of subsystems in the `tcpip` group on the local machine.

6. To get the status of the `tester` subserver, enter:

```
lssrc  -t tester  -p 1234
```

This gets the status of `tester` subserver that belongs to the `srctest` subsystem with the subsystem PID of `1234` on the local machine.

7. To get the status of the subsystem by PID, enter:

```
lssrc  -l  -p 1234
```

This gets the long status of the subsystem with the PID of `1234`.

Files

Item	Description
/etc/objrepos/SRCsubsys	Specifies the SRC Subsystem Configuration Object Class.
/etc/objrepos/SRCsubsvr	Specifies the SRC Subserver Configuration Object Class.
/etc/objrepos/SRCnotify	Specifies the SRC Notify Configuration Object Class.
/etc/services	Defines the sockets and protocols used for Internet services.
/dev/SRC	Specifies the AF_UNIX socket file.
/dev/.SRC-unix	Specifies the location for temporary socket files.

Problem determination commands

When the Topology Services or Group Services subsystem encounter events that require system administrator attention, it uses the First failure data capture (FFDC) facility of RSCT to generate entries in an AIX error log on AIX nodes and the system log on Linux nodes.

ct_ffdc.h File

Purpose

Provides data types, definitions, and interface prototypes for the First Failure Data Capture (FFDC) C language library interfaces.

Description

This header file must be included by any C and C++ language source code files that make use of the First Failure Data Capture C language interfaces. This file contains the C language prototypes for the First Failure Data Capture interfaces, the symbolic constants used as return codes from these interfaces, and data type definitions needed by First Failure Data Capture 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 First Failure Data Capture interfaces should be used during compilation of the source code. Currently, only one version of the First Failure Data Capture interfaces are available, and the value of **FC_VERSION** is set to a default value of 1. Future versions of the First Failure Data Capture interfaces can be used during compilation—when they become available—by setting the value for **FC_VERSION** on the compilation command line. 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 a First Failure Data Capture 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. First Failure Data Capture 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

A First Failure Data Capture client application uses the **fc_init** interface to specify how the FFDC Environment should be established. The following selections are supported:

FC_LOG

A basic FFDC Environment is established, 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 either the **FC_STACK_CREAT** or the **FC_STACK_INHERIT** options described below. 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 wish to use the **fc_push_stack** interface as well as the **fc_log_error** interface. This selection is not to be combined with the **FC_STACK_INHERIT** option described next.

FC_STACK_INHERIT

Inherit 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 combine this selection with the **FC_STACK_CREAT** option specified previously.

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**.

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>
```

Location

/usr/sbin/rsct/include/ct_ffdc.h

/usr/include/rsct/ct_ffdc.h

ctsnap Command

Purpose

Gathers configuration, log, and trace information about the Reliable Scalable Cluster Technology (RSCT) components.

Syntax

ctsnap [**-a**] [**-c** *cluster_name_pattern*] [**-C** *cluster_ID_pattern*] [**-d** *output_dir*] [**-D** *daemon_name_pattern*] [**-k** *stackdump_default*] [**-n** *node_name_pattern*] [**-N** *node_ID_pattern*] [**-p** *days* | { **-f** *from_date* **-t** *to_date* }] [**-s** *spool_dir*] [**-S** *size*] [**-x** *runrpttr*] [**-h**] [**-z**]

Description

The **ctsnap** command gathers configuration, log, and trace information about the RSCT components that are installed with AIX or PowerHA. This command collects data only for the local node on which it is running. Depending on the programs that are installed, information about 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**)
- 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**)
- Management domain resource manager (**IBM.MgmtDomainRM**)
- Microsensor resource manager (**IBM.MicroSensorRM**)
- Recovery resource manager (**IBM.RecoveryRM**)
- Resource monitoring and control (**ctrmc**)
- Sensor resource manager (**IBM.SensorRM**)
- Storage resource manager (**IBM.StorageRM**)
- Topology services (**cthats**)
- Virtual shared disk (**vsd**) (on AIX 6.1)
- Recoverable virtual shared disk (**rvsd**) (on AIX 6.1)

If a problem occurs with any of these components, you can run this command in order to provide information to your software service organization.

The output of the **ctsnap** command consists of a compressed tar file (**ctsnap.***node_name***.***nnnnnnnn***.tar.Z**) and a log file (**ctsnap.***node_name***.***nnnnnnnn***.log**), where *node_name* is the name of the node on which **ctsnap** was run, and *nnnnnnnn* is the time stamp of when the **ctsnap** command was run. Provide both of these files to your software service organization. By default, **ctsnap** puts these files in the **/tmp/ctsupt** directory. Use the **-d** flag to specify a different output directory.

When needed, you can use **ctsnap** to collect information about spooled trace files. Use the **-c**, **-C**, **-D**, **-f**, **-n**, **-N**, **-p**, **-s**, **-S**, and **-t** flags to capture a subset of trace information. You can use the **ctsnap -k stackdump_default** command to produce a stack dump for the following RSCT subsystems:
- Audit log resource manager (**IBM.AuditRM**)
- Common information model resource manager (**IBM.CIMRM**)
- Configuration resource manager (**IBM.ConfigRM**)
- Event response resource manager (**IBM.ERRM**)
- File system resource manager (**IBM.FSRM**)
- Generic resource manager (**IBM.GblResRM**)
- Group services (**cthags**)
- Least-privilege resource manager (**IBM.LPRM**)
- Microsensor resource manager (**IBM.MicroSensorRM**)
- Recovery resource manager (**IBM.RecoveryRM**)
- Resource monitoring and control (**ctrmc**)
- Sensor resource manager (**IBM.SensorRM**)
- Storage resource manager (**IBM.StorageRM**)
- Topology services (**cthats**)

To format the trace file contents of all of the RSCT resource managers, use the **-x** flag.

You can also use the **ctsnap** command to obtain the trace and logging root directory from the RSCT File configuration file (**ctfile.cfg**).

Flags

-a Collects information pertinent only to High Availability Cluster Multi-Processing (HACMP™) clusters on the Linux operating system.

-c *cluster_name_pattern*
Specifies a selection pattern that will limit trace collection to certain cluster names. The pattern is interpreted as a Perl-language regular expression.

-C *cluster_ID_pattern*
Specifies a selection pattern that will limit trace collection to certain cluster IDs. The pattern is interpreted as a Perl-language regular expression.

-d *output_dir*
Specifies the output directory. The default directory is **/tmp/ctsupt**.

-D *daemon_name_pattern*
Specifies a selection pattern that will limit trace collection to certain daemons. The pattern is interpreted as a Perl-language regular expression.

-f *from_date*
Specifies the date from which you want to collect information. The format of the *from_date* parameter is:

```
yyyy-mm-dd[.hh[:mm[:ss]]]
```

Note: Use **-f** in conjunction with the **-t** flag.

-k stackdump_default
Produces a stack dump for these RSCT subsystems: **cthags**, **cthats**, **ctrmc**, **IBM.AuditRM**, **IBM.CIMRM**, **IBM.ConfigRM**, **IBM.ERRM**, **IBM.FSRM**, **IBM.GblResRM**, **IBM.LPRM**, **IBM.MicroSensorRM**, **IBM.RecoveryRM**, **IBM.SensorRM**, and **IBM.StorageRM**.

-n *node_name_pattern*
Specifies a selection pattern that limits the trace collection to certain node names. The pattern is interpreted as a Perl-language regular expression.

-N *node_ID_pattern*
Specifies a selection pattern that limits the trace collection to certain node IDs. The pattern is interpreted as a Perl-language regular expression.

-p *days*
Specifies how many previous days' worth of spooled trace information to collect.

-s *spool_dir*
Captures trace files for the specified spooling directory.

-S *size* Specifies the maximum cumulative size of all of the trace files to collect (in megabytes).

-t *to_date*
Specifies the date to which you want to collect information. The format of the *to_date* parameter is:

```
yyyy-mm-dd[.hh[:mm[:ss]]]
```

Note: Use **-t** in conjunction with the **-f** flag.

-x runrpttr
Formats the trace file contents of all of the RSCT resource managers.

Using this flag increases 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.

-z Prevents collecting the **snap caa** information even in a Cluster Aware AIX (CAA) environment.

Security

Only **root** users can run this command.

Exit Status

0 The command ran successfully.

1 The command was not successful.

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 **rsct.core.utils** fileset for AIX®.

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
   ```
3. To capture all trace files for the **/opt/traces** directory, enter:
   ```
   ctsnap -s /opt/traces
   ```
4. To capture all trace files for the **/opt/traces** directories of the configuration resource manager daemons, enter:
   ```
   ctsnap -s /opt/traces -D '.*ConfigRM.*'
   ```
5. To capture all trace files for the **/opt/traces** directory for the date range `08-28-2008` to `08-29-2008`, enter:
   ```
   ctsnap -s /opt/traces -f 08-28-2008 -t 08-29-2008
   ```
6. To capture all trace files for the **/opt/traces** directory for the previous four days, enter:
   ```
   ctsnap -s /opt/traces -p 4
   ```
7. To capture all trace files for the **/opt/traces** directory for the most recent `50` MB of trace information, enter:
   ```
   ctsnap -s /opt/traces -S 50
   ```

Location

/usr/sbin/rsct/bin/ctsnap
Contains the **ctsnap** command

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 was run.

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 was run.

First failure data capture (FFDC) commands

fccheck Command
Purpose

Performs basic problem determination on the First Failure Data Capture (FFDC) utilities.

Syntax

/usr/sbin/rsct/bin/fccheck [**-q**] | [**-h**]

Description

fccheck 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.

Flags

-h Displays help and usage information to standard output. No other processing is performed.

-q Specified "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**.

Exit Status

The following integer exit status codes can be generated by this command:

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.

fcclear Command
Purpose

Removes 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*]]

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.

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.

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.

fcdecode Command
Purpose

Translates a First Failure Data Capture (FFDC) Failure Identifier from its standard form into its component parts, displaying this information to the standard output device in human readable format.

Syntax

/usr/sbin/rsct/bin/fcdecode *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...] | **-h**

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.

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.

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 1998 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 1998 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.

fcdispfid Command

Purpose

Displays the First Failure Data Capture Failure Identifier (FFDC Failure Identifier) to the standard error device.

Syntax

/usr/sbin/rsct/bin/fcdispfid [**-q**]*FFDC_Failure_ID* | **-h**

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.

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**.

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.

fcfilter Command
Purpose

Locates and displays any First Failure Data Capture (FFDC) Failure Identifiers in a file or in standard input. More than one file may be specified.

Syntax

/usr/sbin/rsct/bin/fcfilter [*file_name*] [. . .]

Description

This commands scans any files listed as arguments for First Failure Data Capture (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.

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.

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.

fcinit Command
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**]

Description

This interface must be used by a script program that wishes to 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.

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 wishes to make use of 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).

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.

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 -l
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.

fclogerr Command

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_syslog_message_text* } | **-h**

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 also 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.

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 One or more data items that provides detailed information on 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 quotes ("). The data items themselves must not contain commas (,), as 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. Current 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 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 error logging 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 error logging 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 LPP'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 programming product in which this software was shipped. This value should be recognizable to both customer and application-support services as an acceptable name for the LPP. Examples of such values are: PSSP, GPFS™, LoadLeveler®, and RSCT. If this option is not provided or appears invalid, 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 Logging template in the error log repository. The **errupdate** command that builds error logging 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 Logging 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 Logging 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.

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.

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 invalid option to this command.

On AIX platforms other than AIX, **fclogerr** returns the following exit status codes when a failure occurs:

38 A record could not be made int he 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, LPP 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 may not be enabled, or may be experiencing problems. 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 Logging 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 example source code segment below 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 Logging 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-quotes 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.

fcpushstk Command
Purpose

Records information about failure or noteworthy conditions to the First Failure Data Capture Error Stack.

Syntax

/usr/sbin/rsct/bin/fcpushstk { [**-a** *assoc_fid*] **-c** *message_catalog_name* **-m** *message_set* **-n** *message_number* [**-o** *message_param* [,*message_param*,...]] **-l** *lpp_name* **-p** *line_of_code_pos* **-r** *resource* **-s** *source_filename* **-v** *sidlevel* {[**-d** *detail_data*] | [**-f** *detail_data_file*]} } *default_message* | **-h**

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.

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, 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.

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. 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.

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, LPP 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.

fcreport Command

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**

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.

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.

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.

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.

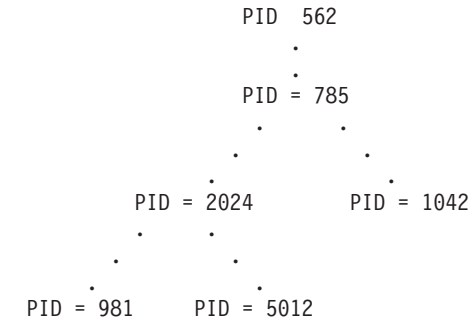
12 Unknown option specified to this command.

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.

Implementation Specifics

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

fcstkrpt Command

Purpose

Displays the contents of an FFDC Error Stack file.

Syntax

/usr/sbin/rsct/bin/fcstkrpt { **[-a]** [**-p** | **-r**] { **-f** *FFDC_Failure_Identifier* [**-i**] | **-s** *FFDC_Error_Stack_File_Name* } } | [**-h**]

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*.

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.

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.

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.

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.562.19981001143052**:

```
$ fcstkrpt -r -s myprog.562.19981001143052
```

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.

fcteststk Command
Purpose

Test for the presence of a First Failure Data Capture Error Stack environment.

Syntax

/usr/sbin/rsct/bin/fcteststk [**-q**] | [**-h**]

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.

Flags

Item	Description
-h	Displays a usage message for this command. No further processing is performed.
-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.

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.

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.

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.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this

one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 903
11501 Burnet Road
Austin, TX 78758-3400
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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Privacy policy considerations

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as the customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at http://www.ibm.com/privacy and IBM’s Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

A

C

D

E

F

G

H

I

L

M

N

P

R

S

T

U

W

X

Printed in USA