IBM Cluster Systems Management for Linux®

Monitoring HOWTO

Version 1 Release 1

SA22-7852-00

IBM Cluster Systems Management for Linux®

Monitoring HOWTO

Version 1 Release 1

SA22-7852-00

Note!
Before using this information and the product it supports, read the information in “Notices” on page 37.

First Edition (June 2001)

This edition of the *Cluster Systems Management for Linux Monitoring HOWTO* applies to Cluster Systems Management for Linux Version 1 Release 1, program number 5799-GNJ, and to all subsequent releases of this product until otherwise indicated in new editions.

IBM® welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

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

IBMLink (United States customers only): IBMUSM10(MHVRCFS)
IBM Mail Exchange: USIB6TC9 at IBMMAIL
Internet e-mail: mhvrcfs@us.ibm.com

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

Make sure to include the following in your comment or note:
- Title and order number of this book
- Page number or topic related to your comment

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

About This HOWTO

This HOWTO describes the Cluster Systems Management for Linux monitoring tool.

Who Should Use This HOWTO

This HOWTO is intended for system administrators responsible for Cluster Systems Management for Linux. The system administrator should have UNIX and networked systems experience. This HOWTO can also be used by system operators or others who need to monitor system status.

How to Use This HOWTO

This HOWTO contains information on how to use the Monitoring application, which is available through the command-line interface. The HOWTO also contains tables of predefined conditions, expressions, and responses that are prepackaged with the application and information on how to configure certain aspects of the system, such as security. A chapter on troubleshooting common problems is also provided.

Typographic Conventions

This HOWTO uses the following typographic conventions:

Typographic	Usage
Bold	• **Bold** words or characters represent system elements that you must use literally, such as commands, flags, and path names.
Italic	• *Italic* words or characters represent variable values that you must supply. • *Italics* are also used for book titles and for general emphasis in text.
`Constant width`	Examples and information that the system displays appear in `constant width` typeface.
[ ]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
|	A vertical bar separates items in a list of choices. (In other words, it means “or.”)
< >	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.
...	An ellipsis indicates that you can repeat the preceding item one or more times.
<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>**.
****	The continuation character is used in coding examples in this book for formatting purposes.

Related Information

- Conserver open source software Web site (http://www.conserver.com)
- IBM Advanced System Management PCI Adapter Firmware Update Diskette
- *IBM Advanced Systems Management PCI Adapter Software user’s guide*
- *Cluster Systems Management for Linux Overview HOWTO*, SA22-7857-00
- *Cluster Systems Management for Linux Remote Control HOWTO*, SA22-7856-00
- *Cluster Systems Management for Linux Set-Up HOWTO*, SA22-7853-00
- *Cluster Systems Management for Linux Technical Reference*, SA22-7851-00
- IBM Linux Clusters Web site (http://www.ibm.com/eserver/clusters/linux)

- *Universal Management Services user's guide*

For information on using serial devices, see these Linux HOWTO documents, located at /usr/doc/HOWTO or on the Linux Documentation Project Web site (http://metalab.unc.edu/mdw/index.html):

- *Serial-HOWTO*
- *Serial-Programming-HOWTO*
- *Modem-HOWTO*

How to Obtain Publications

The Cluster Systems Management for Linux publications are available as HTML and PDF files on the CD-ROM in the /doc directory or on the installed system in the /opt/csm/doc directory. The README is available on the CD-ROM in the root directory (/). The file names are as follows:

- *Cluster Systems Management for Linux Monitoring HOWTO*, csmadm.pdf
- *Cluster Systems Management for Linux Overview HOWTO*, csmovrvw.pdf
- *Cluster Systems Management for Linux Remote Control HOWTO*, csmremot.pdf
- *Cluster Systems Management for Linux Set-Up HOWTO*, csmsetup.pdf
- *Cluster Systems Management for Linux Technical Reference*, csmtech.pdf

Publications for Cluster Systems Management for Linux were also available at the time of this release at the IBM Linux Clusters Web site (http://www.ibm.com/eserver/clusters/linux). The IBM Linux Clusters Web site also includes links to the following resources:

- Conserver open source Web site.
- IBM Netfinity system management utility and documentation Web site.
- Linux Documentation Project HOWTOs Web site.

Contents

Chapter 1. Overview of Cluster Systems Management

The **Cluster Systems Management (CSM)** Monitoring application offers a comprehensive set of monitoring and response capabilities that lets you detect, and in many cases correct, system resource problems such as a critical filesystem becoming full. You can monitor virtually all aspects of your system resources and specify a wide range of actions to be taken when a problem occurs, from simple notification by e-mail to recovery that runs a user-written script. You can specify an unlimited number of actions to be taken in response to an event.

As system administrator, you have a great deal of flexibility in responding to events. You can respond to an event in different ways based on the day of the week and time of day. The following are some examples of how you can use monitoring:

- You can be alerted by e-mail if **/tmp** is unmounted during working hours, but you can have the problem logged if **/tmp** is unmounted during nonworking hours.
- You can be notified by e-mail when **/var** is 80% full.
- You can have a user-written script run automatically to delete the oldest unnecessary files when **/tmp** is 90% full.

CSM uses Resource Monitoring and Control (RMC) to monitor the system and to perform many of its operations. For information about the command line interface to the RMC subsystem, see *Cluster Systems Management for Linux Technical Reference*. For information on RMC diagnostic information, see "Recovering from RMC and Resource Manager Problems" on page 33. For authorization and modifying the ACL file, see "Security Considerations" on page 4.

Monitoring Concepts

Monitoring lets you detect conditions of interest in the cluster nodes and their associated resources and automatically take action when those conditions occur. The key elements in monitoring are conditions and responses. A condition identifies one or more resources that you want to monitor, such as the **/var** file system, and the specific resource state you are interested in, such as /var>90% full. A response specifies one or more actions to be taken when the condition is found to be true. Actions can include notification, running commands, and logging.

About Conditions

System resources that you can monitor are organized into general categories called *resource classes*. Examples of resource classes include Processor, File System, Physical Volume, and Ethernet Device.

Each resource class includes individual system resources that belong to the class. For example, the File System resource class might include these resources:

- **/tmp**
- **/var**
- **/usr**
- **/home**

When a resource is specified for use in a condition, it is called a *monitored resource*.

Each resource within a resource class also has a set of attributes that you can monitor. For example, the File System resource class has the following attributes available for monitoring:

- OpState - the operational state of the file system (mounted or unmounted).
- PercentTotUsed - the percentage of total file system space that is in use.
- PercentINodeUsed - the percentage of i-nodes in use for the file system.

For a condition, you specify the monitored attribute of the resource in a logical expression that defines a threshold or state of the monitored resource. When the logical expression is true (the threshold is reached or the state becomes true), an event is generated. The logical expression is the *event expression* of the condition. Event expressions are typically used to monitor potential problems and significant changes in the system. For example, the event expression for a /var space used condition might be PercentTotUsed > 90.

The *rearm expression* of a condition is optional. A rearm expression typically indicates when the monitored resource has returned to an acceptable state. When the rearm expression is met, monitoring for the condition resumes. If a rearm event is not specified, when the event expression becomes true an event is generated for certain attributes every time the monitored attribute is evaluated.

If a rearm expression is specified, evaluation of the rearm expression starts after the event expression becomes true. When the rearm expression becomes true, a rearm event is generated; then the evaluation of the event expression starts again. For example, if the event expression for a /var space used condition is 90% full and the rearm expression is PercentTotUsed < 80, then an event is generated when /var is more than 90% full. The next time the condition is evaluated, the rearm expression is used. When /var is less than 80% full, an event is generated indicating that the condition has been reset, and the event expression is used again to evaluate the condition.

See “Using Expressions” on page 11 for more information about data types and operators that you can use in an event expression or a rearm expression.

Predefined conditions are provided with the Monitoring application. To create a new condition, set the following condition components:

Condition Component	Description	Example
Condition name	Required. The name you want to give the condition.	/var space used
Resource class	Required. The resource class to be monitored.	FileSystem
Monitored attribute	Optional. The attribute of the resource class to be monitored. If not specified, it will be extracted from the Event expression.	PercentTotUsed
Monitored resources	Optional. The specific resources in the resource class that are to be monitored. If not specified, the default is all resources in the specified Resource Class.	/var
Event expression	Required. A logical expression defining the value or state of the monitored property that is to generate an event.	PercentTotUsed > 90
Event description	Optional. A text description of the event expression. If not specified, the default is a NULL string.	An event occurs when /var is more than 90% full.

Rearm expression	Optional. When a rearm expression is specified, the rearm expression is evaluated when the event expression becomes true. When the rearm expression becomes true, the event expression is used for evaluation again. If not specified, this condition will only be monitored with the Event expression.	PercentTotUsed < 80
Rearm description	Optional. A text description of the rearm expression. If not specified, the default is a NULL string.	A rearm event occurs when /var is less than 80% full.
Severity	Optional. The severity of the condition: Informational, Warning, or Critical. If not specified, the default is Informational.	Critical

Finally, a user-defined sensor can be created to monitor an attribute of interest. Then expressions can be defined that contain conditions and responses with associated actions to be performed when the attribute has a certain value. For example, a script can be written to return the number of users logged on, and a condition and response can be defined so that a specified action is taken when the number of users exceeds a certain threshold.

About Responses

A response consists of one or more actions to be performed by the system when an event or rearm event occurs for a condition. In the Monitoring application you can use predefined responses or create new responses and associate them with conditions as needed. You can associate multiple responses with one condition, and one response with multiple conditions.

The responses for a condition remain deactivated until you start monitoring for that condition. When you select a condition to start monitoring, you need to activate at least one of its responses. The responses that are not active remain available to be used at another time. This allows you to use different responses for a condition as needed, without having to redefine them.

Predefined responses are provided with the Monitoring application. To create a new response you will need to set the following response and action components:

Response Component	**Description**	**Example**
Response name	The name you want to give the response.	Response for critical conditions
Actions	One or more actions to be taken as part of the response.	Log events to a file

Action Component	**Description**	**Example**
Action name	The name of an action to be taken as part of the response.	Send email to the operator
When in effect	The days and times when this action is to be used to respond to the condition.	08:00 – 17:00 Monday – Friday
Use for event, rearm event, or both	Whether the action is to be used to respond to an event, a rearm event, or both.	Event

Command	The command to be run when an event or rearm event occurs.	A recovery script

You can associate multiple responses with a condition if you want to define different responses based on when the event occurs. For example, you might have a work day response and a weekend response, each containing one or more actions. Consider how you might respond to a /var space used condition with the following responses. During working hours, you might want to email the operator, run a command, and broadcast a message to users who are logged on. During weekend hours, you might want to email the system administrator and log a message to a file.

How Conditions and Responses Work Together

After monitoring for the condition begins, the system evaluates the event expression to see if it is true. When the event expression becomes true, an *event* occurs that automatically notifies all of the associated event responses, which causes each event response to run its defined actions.

The event expression and the rearm expression work together as follows when a condition is monitored. First, the event expression is evaluated. When the event expression becomes true, an event occurs, and the specified actions are taken. When the event expression becomes true, the system begins evaluating the rearm expression. When the rearm expression becomes true, the *rearm event* occurs, which automatically starts the actions defined for the rearm event. When the rearm event occurs, the system returns to evaluating the event expression.

Security Considerations

The mechanisms for authentication and authorization that are provided by CSM are described in the following sections.

Authentication

CSM provides authentication using the Ident protocol. The daemon **identd** listens for TCP connections on a known TCP port 113. Application servers need to connect to this daemon on the host where the client is running. The servers have to provide **identd** with the local and remote ports. The daemon then returns the identity of the owner of the process connected to the remote port, if it exists. The application servers can then use this identity as the remote client's Unix identity.

The security infrastructure assumes that **identd** is running and listening on Port 113. Red Hat Linux includes **identd**. The **identd** code can be downloaded from one of the following sites:

- ftp://ftp.lysator.liu.se/pub/ident/servers/
- http://www2.lysator.liu.se/~pen/pidentd/

identd needs to be started from /etc/rc.d/init.d.

The **/etc/services** file should contain the following:

```
auth        113/tcp        authentication tap ident
```

The **/etc/identd.conf** file should contain the following comment:

```
#-- Disable username lookups (only return uid numbers)
#result:uid-only = no
```

Authorization

CSM provides authorization in the form of an access-control-list (ACL) file. You can create an ACL file to apply access control to resource classes. If you do not create an ACL file, then the system uses the following default permissions:

```
OTHER
   root@LOCALHOST      *   rw
   LOCALHOST           *   r
```

The ACL file is in stanza format. Each stanza begins with the stanza name, which is the name of a resource class. A stanza with the name of OTHER applies to all resource classes that are not otherwise specified in the file.

Each line of the stanza contains a user identifier, an object type, and an optional set of permissions. A stanza line indicates that the user at the host has the permissions to access the resource class or resource instances (or both) for the resource class named by the stanza. The user identifier can have one of the following three forms:

1. *user_name@host_name*
2. *host_name*
3. *

A *host_name* is a fully qualified host domain name or the keyword LOCALHOST. The first form specifies a user running a Resource Monitoring and Control (RMC) application on the named host. If the host name is the keyword LOCALHOST, then the application is running on the same node as the RMC subsystem. The second form specifies any user running an RMC application on the named host. The third form specifies any user running an RMC application on any host.

The object type is one of the characters C, R or *. The letter C indicates that the permissions provide access to the resource class. The letter R indicates that the permissions provide access to all of the resource instances of the class. The asterisk indicates that the permissions provide access to both the resource class and all resource instances of the class.

The permissions provided are represented by one, both, or none of the characters **r** and **w**. The letter **r** indicates that the specified user at the specified host has read permission. The letter **w** indicates that the specified user at the specified host has write permission. Both letters indicate the user has read and write permission. If the permissions are omitted, then the user does not have access to the objects specified by the type character. Read permission allows you to register and unregister for events, to query attribute values, and to validate resource handles. Write permission allows you to run all other command interfaces. Note that no permissions are needed to query resource class and attribute definitions.

For any command issued against a resource class or its instances, the RMC subsystem examines the lines of the stanza matching the specified class in the order specified in the ACL file. The first line that contains 1) an identifier that matches the user issuing the command and 2) an object type that matches the objects specified by the command is the line used to determine access permissions. Therefore, lines containing more specific user identifiers and object types should be placed before lines containing less specific user identifiers and object types.

How to Create and Modify the ACL File

A sample ACL file is provided in **/usr/sbin/rsct/cfg/ctrmc.acls**. This file contains the following default permissions:

```
OTHER
   root@LOCALHOST      *   rw
   LOCALHOST           *   r
```

To change these defaults, you must copy the sample ACL file to **/var/ct/cfg/ctrmc.acls** and put your modifications in that file (or you can create a new ACL file with the same name and location). Then to activate your new permissions, type:

```
refresh -s ctrmc
```

Provided there are no errors in the modified ACL file, the permissions will take effect. If errors are found in the modified ACL file, they are logged to **/var/ct/IW/log/mc/default**.

Examples of ACL File Stanzas

The following examples show ways the ACL file can be modified.

1. For resource class Class_A, user1 at sys1 has permission to read and write all resource instances, and root at **sys1** has permission to read and write both the resource class and all of its resource instances. All other users at **sys1** have permission to read both the resource class and all of its instances. Note that this gives user1 permission to read the resource class.

   The user1 at **sys3** has permission to read and write the resource class; user2 at **sys3** has no permission to access either the resource class or its instances. All other users at **sys3** have permission to read both the resource class and all of its instances.

   Note: If the line containing user2's user ID and the following line were positionally reversed, then the line containing user2's ID would be rendered ineffective.

   Finally, root on the machine containing the ACL file can read and write both the resource class and all of its resource instances.

   ```
   Class_A
      user1@sys1.pok.ibm.com   R    rw
      root@sys1.pok.ibm.com    *    rw
      sys1.pok.ibm.com         *   r
      user1@sys3.pok.ibm.com   C    rw
      user2@sys3.pok.ibm.com   *
      sys3.pok.ibm.com         *    r
      root@LOCALHOST           *    rw
   ```

2. For Class_B, root on the machine containing the ACL file can read and write both the resource class and all of its resource instances. All other users on all hosts can read both the resource class and all of its resource instances.

   ```
   Class_B
      root@LOCALHOST     *   rw
           *             *   r
   ```

3. For all other resource classes (represented by OTHER), root at **sys1** has permission to read both the resource class and all of its resource instances, and root on the machine containing the ACL file can read and write both the resource class and all of its resource instances.

   ```
   OTHER
      root@sys1.pok.ibm.com      *   r
      root@LOCALHOST             *   rw
   ```

ACL File Stanza Syntax

A stanza begins with a line containing the stanza name, which must start in column 1. A stanza line consists of leading white space (one or more blanks and/or tabs, followed by one or more white-space-separated tokens. Comments may be present in the file. Any line in which the first non-white-space character is a pound sign (#) is a comment. Blank lines are also considered comment lines and are ignored. Any part of a line that begins with two consecutive forward slash characters (//), not surrounded by double quotes (″), is considered to be a comment from that point through the end of the line. The stanza lines in an ACL file each contain two or three tokens:

```
stanza_name
     user_identifier    type    permissions
     user_identifier    type    permissions
     |       |
     |       |
     user_identifier    type    permissions
```

The permissions token may be omitted.

For a complete description of the Resource Monitoring and Control components and how to use them, see "Chapter 3. Components Provided for Monitoring" on page 19.

Chapter 2. Using the Monitoring Application

This chapter describes planning for monitoring your system, tracking system events, and using and modifying the predefined scripts, expressions, commands, and responses packaged with this application. These predefined elements and how to use them are described in detail in “Chapter 3. Components Provided for Monitoring” on page 19.

Planning What to Monitor in Your System

First, select conditions to monitor that would have a severe impact on your system. These conditions might include:

- /var space used
- Percentage of free paging space

When you have determined the resource problems you want to monitor, review the predefined conditions and identify the conditions you want to use. Use the **lscondition** command to view all conditions. If a predefined condition deviates from your requirements in some way, you can edit it, use it as a template to create your own customized condition, or create your own condition using the **mkcondition** command. After you have selected conditions for monitoring, you need to plan one or more responses to be taken for the event and the optional rearm event.

Planning How to Respond to Detected Conditions

A set of predefined responses comes installed with your system (see “Predefined Responses” on page 29). Each response has one or more actions associated with it. Each action can be activated or deactivated to fit your particular work environment and schedule.

The predefined actions are:

- Sending email to a particular user (see the **notifyevent** command man page).
- Logging to a user-specified log file (see the **logevent** command man page).
- Broadcasting a message to all users who are currently logged on (see the **wallevent** command man page).
- Displaying a window on an X-window display (see the **displayevent** command man page).
- Sending a message to specified users by means of the **write** command (see the **msgevent** command man page).

You can also write your own commands that correct or mitigate conditions, and run them through the **Run program** option.

You might specify different actions based on when the monitored condition occurs. For example, you could have one set of actions to respond to a condition during working hours and another set to respond to a condition on nights and weekends. To be notified of events when you are away from your terminal, your actions must include email, broadcasting, or logging.

Getting Started with the Monitoring Application

This section describes how to start using the Monitoring application. You can use the command line to do the following:

- Start monitoring your system.
- Associate a response with a condition.
- View events that have occurred on your system.
- Stop monitoring.

How to Associate a Response with a Condition

To associate a response with a condition, use the following commands (see the command man pages or *Cluster Systems Management for Linux Technical Reference* for detailed usage information):

1. Use the **lsresponse** command to list all responses.
2. Use the **mkcondresp** command to associate a condition with a response without starting monitoring.
3. Use the **startcondresp** command to associate a condition with a response and begin monitoring immediately.

How to View Events

To view events use the **lsaudrec** command to view the audit log. You can use the **notifyevents** predefined script to log events to a file.

How to Stop Monitoring

To stop monitoring use the **stopcondresp** command.

How to Monitor Your System Using the Command Line Interface

The following scenarios demonstrate most frequently performed monitoring tasks. See the *Cluster Systems Management for Linux Technical Reference* or the command man pages for detailed usage information.

1. To list the conditions in your system, type: **lscondition**. Output is similar to:

```
Name                          Monitoring Status
"/tmp space used"             "Not monitored"
"var space used"              "Monitored"
(more conditions listed...)
```

2. To list the responses available in the system, type: **lsresponse**. Output is similar to:

```
Name
"Critical notification"
"Warning notification"
"Informational notification"
"Remove unwanted files"
(more responses listed...)
```

3. To list responses associated with a condition, use the **lscondresp** command. For example, to list the responses associated with the condition ″/tmp space used″, type: **lscondresp ″/tmp space used″**. Output is similar to:

```
Condition           Response                    State
"/tmp space used"   "Broadcast event on-shift"  Active
"/tmp space used"   "E-mail root anytime"       Not Active
```

4. To start monitoring a condition, one or more responses need to be specified for the condition. For example, to start monitoring the condition ″/tmp space used″ using the response ″critical notification″ and ″remove unwanted files,″ type:

```
startcondresp "/tmp space used" "critical notification" "remove unwanted files"
```

5. You can either stop monitoring a condition completely, or stop monitoring a condition with specific responses. To stop monitoring the condition ″/tmp space used″ completely, type:

```
stopcondresp "/tmp space used"
```

   To stop monitoring the condition ″/tmp space used″ with a specific response, ″critical notification,″ type:

```
stopcondresp "/tmp space used" "critical notification"
```

6. You can copy a condition to use as a template for a new condition. For example, to create a new condition ″my test condition″ from an existing condition ″/tmp space used,″ type:
   ```
   mkcondition -c "/tmp space used" "my test condition"
   ```

7. To view events or the actions taken in response to the events, type:
   ```
   lsaudrec -l
   ```

For a complete list of predefined commands, scripts, and utilities, see “Predefined Commands, Scripts, Utilities, and Files” on page 30.

Tracking Monitoring Activity

For information about monitoring events, rearm events, actions, and errors that have occurred, view the audit log by using the **lsaudrec** command. See the **lsaudrec** man page and “Using the Audit Log to Track Monitoring Activity” for details.

Using the Audit Log to Track Monitoring Activity

Audit log records include the following:

- Instances of starting and stopping monitoring
- Events and rearm events
- Actions taken in response to events and rearm events
- Results (successful or unsuccessful) of actions taken in response to events and rearm events
- Subsystem errors or monitoring errors

The administrator can use the audit log to track activity that may not be visible otherwise because the activity is related to subsystems running in the background. To list audit log records, use the **lsaudrec** command. To remove audit log records, use the **rmaudrec** command. For details see the command man pages, or *Cluster Systems Management for Linux Technical Reference*.

Using Scripts

The *Cluster Systems Management for Linux Technical Reference* contains information about predefined scripts that are provided with the Event Response resource manager (ERRM). The following scripts are provided:

- **displayevent**
- **logevent**
- **msgevent**
- **notifyevent**
- **wallevent**

You can also use existing operating system commands and user-written scripts in the definition of an action.

Using Predefined Response Scripts

The **displayevent**, **logevent**, **msgevent**, **notifyevent**, and **wallevent** scripts are examples of the types of actions that system administrators can use to respond to events. The **displayevent** script displays an event or a rearm event to a specified X-window display. The **logevent** script appends a formatted string containing the specifics of an event to a user-specified text file. The **msgevent** script sends an event or a rearm event to a specified user’s console. The **notifyevent** script captures the event information and sends the event information via UNIX mail to a specified userid. The **wallevent** script broadcasts a message to all users who are logged in. For a full description of these scripts, see *Cluster Systems Management for Linux Technical Reference* or the command man pages.

You can use these scripts as-is or treat them as templates by copying and modifying them to create new scripts that suit your needs. For example, to use the **wallevent** script as a template for a page event command, do the following:

1. Copy the **wallevent** script at **/usr/sbin/rsct/bin/wallevent** to a new script file and rename it, for example, to **pageevent**.
2. Replace the **wall** command with the program for your pager.

For a command to run in response to an event or a rearm event defined by a condition, the command must be included as an action in an Event Response resource. When an Event Response resource is defined, specify the entire path name for a script that is used within an action. Use the Event Response resource manager commands to set up responses.

Test any scripts or commands that you have created or modified before you use them as actions in a production environment.

Using Event Response Environment Variables

After ERRM has subscribed to RMC to monitor a condition and that condition occurs, the ERRM runs commands in the user's operating system environment. The Event Response resource contains a list of commands to be run. Before each command is run, the following environment variables are established for the command to use (see "Event Response Resource Manager" on page 21 for a detailed description of the ERRM):

- **ERRM_ATTR_NAME** - The display name of the dynamic attribute used in the expression that caused this event to occur. (A variable name is restricted to include only 7–bit ASCII characters that are alphanumeric (a-z, A-Z, 0-9) or the underscore character (_). The name must begin with an alphabetic character.)
- **ERRM_COND_HANDLE** - The Condition resource handle (six hexadecimal integers that are separated by spaces and written as a string) that caused the event.
- **ERRM_COND_NAME** - The name of the Condition resource that caused the event.
- **ERRM_COND_SEVERITY** - The significance of the Condition resource that caused the event. For the severity attribute values of 0, 1, and 2, this environment variable has the following values respectively: informational, warning, critical. All other Condition resource severity attribute values are represented in this environment variable as a decimal string.
- **ERRM_DATA_TYPE** - RMC ct_data_type_t of the dynamic attribute that changed to cause this event. The following is a list of valid values for this environment variable: CT_INT32, CT_UINT32, CT_INT64,CT_UINT64, CT_FLOAT32, CT_FLOAT64, CT_CHAR_PTR, CT_BINARY_PTR, and CT_SD_PTR. For all data types except CT_NONE, the ERRM_VALUE of the environment variable is defined with the value of the dynamic attribute.
- **ERRM_ER_HANDLE** - The Event Response resource handle (six hexadecimal integers that are separated by spaces and written as a string) for this event.
- **ERRM_ER_NAME** - The name of the Event Response resource that is running this command.
- **ERRM_EXPR** - The expression that was evaluated which caused the generation of this event. This could be either the event or rearm expression, depending on the type of event that occurred. This can be determined by the value of ERRM_TYPE.
- **ERRM_NODE_NAME** - The host name on which this event or rearm event occurred.
- **ERRM_RSRC_CLASS_NAME** - The display name of the resource class of the dynamic attribute that caused the event to occur.
- **ERRM_RSRC_HANDLE** - The resource handle of the resource whose state change caused the generation of this event (written as a string of six hexadecimal integers that are separated by spaces).
- **ERRM_RSRC_NAME** - The name of the resource whose dynamic attribute changed to cause this event.

- **ERRM_SD_DATA_TYPES** - The data type for each element within the structured data (SD) variable separated by commas. This environment variable is only defined when ERRM_DATA_TYPE is CT_SD_PTR. For example: CT_CHAR_PTR, CT_UINT32_ARRAY, CT_UINT32_ARRAY, CT_UINT32_ARRAY
- **ERRM_TIME** - The time the event occurred written as a decimal string that represents the time since midnight January 1, 1970 in seconds, followed by a comma and the number of microseconds.
- **ERRM_TYPE** - The type of event that occurred. The two possible values for this environment variable are: event or rearm event.
- **ERRM_VALUE** - The value of the dynamic attribute that caused the event to occur for all dynamic attributes except those with a data type of CT_NONE.

  The following data types are represented with this environment variable as a decimal string: CT_INT32, CT_UINT32, CT_INT64, CT_UINT64, CT_FLOAT32, and CT_FLOAT64.

  CT_CHAR_PTR is represented as a string for this environment variable.

  CT_BINARY_PTR is represented as a hexadecimal string separated by spaces.

  CT_SD_PTR is enclosed in square brackets and has individual entries within the SD that are separated by commas. Arrays within an SD are enclosed within braces {}. For example, [″My Resource Name″,{1,5,7},{0,9000,20000},{7000,11000,25000}] See the definition of ERRM_SD_DATA_TYPES for an explanation of the data types that these values represent.

(See ″Resource Handle″ on page 13 for a definition and an example of a resource handle.)

Using Expressions

The information in this section is for advanced users who want to:

- modify predefined expressions
- select resources
- filter audit log records by compiling and running a complex mathematical expression against a set of values

Permissible data types and operators are described, and the order of precedence for the operators is included. RMC uses these functions to match a selection string against the persistent attributes of a resource and to implement the evaluation of an event expression or a rearm expression.

An expression is similar to a C language statement or the WHERE clause of an SQL query. It is composed of variables, operators, and constants. The C and SQL syntax styles may be intermixed within a single expression. The following table relates the RMC terminology to SQL terminology:

RMC	SQL
attribute name	column name
select string	WHERE clause
operators	predicates, logical connectives
resource class	table

SQL Restrictions

SQL syntax is supported for selection strings, with the following restrictions:

- Only a single table may be referenced in an expression.
- Queries may not be nested.
- The IS NULL predicate is not supported because there is no concept of a NULL value.
- The period (.) operator is not a table separator (for example, table.column). Rather, in this context, the period (.) operator is used to separate a field name from its containing structure name.

- The pound sign (#) is hard-coded as the escape character within SQL pattern strings.
- All column names are case sensitive.
- All literal strings must be enclosed in either single or double quotation marks. Bare literal strings are not supported because they cannot be distinguished from column and attribute names.

Supported Base Data Types

The term variable is used in this context to mean the column name or attribute name in an expression. Variables and constants in an expression may be one of the following data types that are supported by the RMC subsystem:

Symbolic Name	Description
CT_INT32	Signed 32–bit integer
CT_UINT32	Unsigned 32–bit integer
CT_INT64	Signed 64–bit integer
CT_UINT64	Unsigned 64–bit integer
CT_FLOAT32	32–bit floating point
CT_FLOAT64	64–bit floating point
CT_CHAR_PTR	Null-terminated string
CT_BINARY_PTR	Binary data – arbitrary-length block of data
CT_RSRC_HANDLE_PTR	Resource handle – an identifier for a resource that is unique over space and time (20 bytes)

Structured Data Types

In addition to the base data types, aggregates of the base data types may be used as well. The first aggregate data type is similar to a structure in C in that it can contain multiple fields of different data types. This aggregate data type is referred to as *structured data* (SD). The individual fields in the structured data are referred to as *structured data elements* or simply *elements*. Each element of a structured data type may have a different data type which can be one of the base types in the preceding table or any of the array types discussed in the next section, except for the structured data array.

The second aggregate data type is an array. An array contains zero or more values of the same data type, such as an array of CT_INT32 values. Each of the array types has an associated enumeration value (CT_INT32_ARRAY, CT_UINT32_ARRAY). Structured data may also be defined as an array but is restricted to have the same elements in every entry of the array.

Data Types That Can Be Used for Literal Values

Literal values can be specified for each of the base data types as follows:

Array An array or list of values may be specified by enclosing variables or literal values, or both, within braces {} or parentheses () and separating each element of the list with a comma. For example: { 1, 2, 3, 4, 5 } or (″abc″, ″def″, ″ghi″)

Entries of an array can be accessed by specifying a subscript as in the C programming language. The index corresponding to the first element of the array is always zero; for example, List [2] references the third element of the array named List. Only one subscript is allowed. It may be a variable, a constant, or an expression that produces an integer result. For example, if List is an integer array, then List[2]+4 produces the sum of 4 and the current value of the third entry of the array.

Binary Data

A binary constant is defined by a sequence of hexadecimal values, separated by white space. All hexadecimal values comprising the binary data constant are enclosed in double quotation marks.

Each hexadecimal value includes an even number of hexadecimal digits, and each pair of hexadecimal digits represents a byte within the binary value. For example:

```
"0xabcd 0x01020304050607090a0b0c0d0e0f1011121314"
```

Character Strings

A string is specified by a sequence of characters surrounded by single or double quotation marks (you can have any number of characters, including none). Any character may be used within the string except the null '\0' character. Double quotation marks and backslashes may be included in strings by preceding them with the backslash character.

Floating Types

These types can be specified by the following syntax:

- A leading plus (+) or minus (-) sign
- One or more decimal digits
- A radix character, which at this time is the period (.) character
- An optional exponent specified by the following:
  - A plus (+) or minus (-) sign
  - The letter 'E' or 'e'
  - A sequence of decimal digits (0–9)

Integer Types

These types can be specified in decimal, octal, or hexadecimal format. Any value that begins with the digits 1–9 and is followed by zero or more decimal digits (0–9) is interpreted as a decimal value. A decimal value is negated by preceding it with the character '-'. Octal constants are specified by the digit 0 followed by 1 or more digits in the range 0–7. Hexadecimal constants are specified by a leading 0 followed by the letter x (uppercase or lowercase) and then followed by a sequence of one or more digits in the range 0–9 or characters in the range a–f (uppercase or lowercase).

Resource Handle

A fixed-size entity that consists of two 16–bit and four 32–bit words of data. A literal resource handle is specified by a group of six hexadecimal integers. The first two values represent 16–bit integers and the remaining four each represent a 32–bit word. Each of the six integers is separated by white space. The group is surrounded by double quotation marks. The following is an example of a resource handle:

```
"0x4018 0x0001 0x00000000 0x0069684c 0x00519686 0xaf7060fc"
```

Structured Data

Structured data values can be referenced only through variables. Nevertheless, the RMC command-line interface displays structured data (SD) values and accepts them as input when a resource is defined or changed. A literal SD is a sequence of literal values, as defined in "Data Types That Can Be Used for Literal Values" on page 12, that are separated by commas and enclosed in square brackets. For example, ['abc',1,{3,4,5}] specifies an SD that consists of three elements: (a) the string 'abc', (b) the integer value 1, and (c) the three-element array {3,4,5}.

Variable names refer to values that are not part of the expression but are accessed while running the expression. For example, when RMC processes an expression, the variable names are replaced by the corresponding persistent or dynamic attributes of each resource.

Entries of an array may be accessed by specifying a subscript as in 'C'. The index corresponding to the first element of the array is always 0 (for example, List[2] refers to the third element of the array named List). Only one subscript is allowed. It may be a variable, a constant, or an expression that produces an integer result. A subscripted value may be used wherever the base data type of the array is used. For example, if List is an integer array, then ″List[2]+4″ produces the sum of 4 and the current value of the third entry of the array.

The elements of a structured data value can be accessed by using the following syntax:

```
<variable name>.<element name>
```

For example, `a.b`

The variable name is the name of the table column or resource attribute, and the element name is the name of the element within the structured data value. Either or both names may be followed by a subscript if the name is an array. For example, `a[10].b` refers to the element named b of the 11th entry of the structured data array called a. Similarly, `a[10].b[3]` refers to the fourth element of the array that is an element called b within the same structured data array entry `a[10]`.

How Variable Names Are Handled

Variable names refer to values that are not part of an expression but are accessed while running the expression. When used to select a resource, the variable name is a persistent attribute. When used to generate an event, the variable name is a dynamic attribute. When used to select audit records, the variable name is the name of a field within the audit record.

A variable name is restricted to include only 7–bit ASCII characters that are alphanumeric (a–z, A–Z, 0–9) or the underscore character (_). The name must begin with an alphabetic character. When the expression is used by the RMC subsystem for an event or a rearm event, the name can have a suffix that is the '@' character followed by 'P', which refers to the previous observation.

Operators That Can Be Used in Expressions

Constants and variables may be combined by an operator to produce a result that in turn may be used with another operator. The resulting data type or the expression must be a scalar integer or floating-point value. If the result is zero, the expression is considered to be FALSE; otherwise, it is TRUE.

Note: Blanks are optional around operators and operands unless their omission causes an ambiguity. An ambiguity typically occurs only with the word form of operator (that is, AND, OR, IN, LIKE, etc.). With these operators, a blank or separator, such as a parenthesis or bracket, is required to distinguish the word operator from an operand. For example, aANDb is ambiguous. It is unclear if this is intended to be the variable name aANDb or the variable names a, b combined with the operator AND. It is actually interpreted by the application as a single variable name aANDb. With non-word operators (for example, +, -, =, &&, etc.) this ambiguity does not exist, and therefore blanks are optional.

The set of operators that can be used in strings is summarized in the following table:

Operator	Description	Left Data Types	Right Data Types	Example	Notes
+	Addition	Integer,float	Integer,float	″1+2″ results in 3	None
-	Subtraction	Integer,float	Integer,float	″1.0-2.0″ results in -1.0	None
*	Multiplication	Integer,float	Integer,float	″2*3″ results in 6	None
/	Division	Integer,float	Integer,float	″2/3″ results in 1	None
-	Unary minus	None	Integer,float	″-abc″	None
+	Unary plus	None	Integer,float	″+abc″	None
..	Range	Integers	Integers	″1..3″ results in 1,2,3	Shorthand for all integers between and including the two values
%	Modulo	Integers	Integers	″10%2″ results in 0	None
|	Bitwise OR	Integers	Integers	″2|4″ results in 6	None
&	Bitwise AND	Integers	Integers	″3&2″ results in 2	None

Operator	Description	Left Data Types	Right Data Types	Example	Notes
~	Bitwise complement	None	Integers	~0x0000ffff results in 0xffff0000	None
^	Exclusive OR	Integers	Integers	0x0000aaaa^0x0000ffff results in 0x00005555	None
>>	Right shift	Integers	Integers	0x0fff>>4 results in 0x00ff	None
<<	Left shift	Integers	Integers	″0x0ffff<<4″ results in 0xffff0	None
== =	Equality	All but SDs	All but SDs	″2==2″ results in 1 ″2=2″ results in 1	Result is true (1) or false (0)
!= <>	Inequality	All but SDs	All but SDs	″2!=2″ results in 0 ″2<>2″ results in 0	Result is true (1) or false (0)
>	Greater than	Integer,float	Integer,float	″2>3″ results in 0	Result is true (1) or false (0)
>=	Greater than or equal	Integer,float	Integer,float	″4>=3″=1	Result is true (1) or false (0)
<	Less than	Integer,float	Integer,float	″4<3″ results in 0	Result is true (1) or false (0)
<=	Less than or equal	Integer,float	Integer,float	″2<=3″ results in 1	Result is true (1) or false (0)
=~	Pattern match	Strings	Strings	″abc″=~″a.*″ results in 1	Right operand is interpreted as an extended regular expression
!~	Not pattern match	Strings	Strings	″abc″!~″a.*″ results in 0	Right operand is interpreted as an extended regular expression
=? LIKE like	SQL pattern match	Strings	Strings	″abc″=? ″a%″ results in 1	Right operand is interpreted as a SQL pattern
!? NOT LIKE not like	Not SQL pattern match	Strings	Strings	″abc″!? ″a%″ results in 0	Right operand is interpreted as a SQL pattern
|< IN in	Contains any	All but SDs	All but SDs	″{1..5}|<{2,10}″ results in 1	Result is true (1) if left operand contains any value from right operand

Operator	Description	Left Data Types	Right Data Types	Example	Notes
>< NOT IN not in	Contains none	All but SDs	All but SDs	″{1..5}><{2,10}″ results in 1	Result is true (1) if left operand contains no value from right operand
&<	Contains all	All but SDs	All but SDs	″{1..5}&<{2,10}″ results in 0	Result is true (1) if left operand contains all values from right operand
|| OR or	Logical OR	Integers	Integers	″(1<2)||(2>4)″ results in 1	Result is true (1) or false (0)
&& AND and	Logical AND	Integers	Integers	″(1<2)&&(2>4)″ results in 0	Result is true (1) or false (0)
! NOT not	Logical NOT	None	Integers	″!(2==4)″ results in 1	Result is true (1) or false (0)

When integers of different signs or size are operands of an operator, standard C style casting is implicitly performed. When an expression with multiple operators is evaluated, the operations are performed in the order defined by the precedence of the operator. The default precedence can be overridden by enclosing the portion or portions of the expression to be evaluated first in parentheses (). For example, in the expression ″1+2*3″, multiplication is normally performed before addition to produce a result of 7. To evaluate the addition operator first, use parentheses as follows: ″(1+2)*3″. This produces a result of 9. The default precedence rules are shown in the following table. All operators in the same table cell have the same or equal precedence.

Operators	Description
.	Structured data element separator
~ ! NOT not - +	Bitwise complement Logical not Unary minus Unary plus

*	Multiplication
/	Division
%	Modulo
+	Addition
-	Subtraction
<<	Left shift
>>	Right shift
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
==	Equality
!=	Inequality
=?	SQL match
LIKE	
like	
!?	SQL not match
=~	Reg expr match
!~	Reg expr not match
?=	Reg expr match (compat)
|<	Contains any
IN	
in	
><	Contains none
NOT IN	
not in	
&<	Contains all
&	Bitwise AND
^	Bitwise exclusive OR
|	Bitwise inclusive OR
&&	Logical AND
||	Logical OR
,	List separator

Pattern Matching

Two types of pattern matching are supported; extended regular expressions and that which is compatible with the standard SQL LIKE predicate. This type of pattern may include the following special characters:

- The percentage sign (%) matches zero or more characters.
- The underscore (_) matches exactly one character.
- All other characters are directly matched.
- The special meaning for the percentage sign and the underscore character in the pattern may be overridden by preceding these characters with an escape character, which is the pound sign (#) in this implementation.

Examples of Expressions

Some examples of the types of expressions that can be constructed follow:

1. The following expressions match all rows or resources that have a name which begins with 'tr' and ends with '0', where 'Name" indicates the column or attribute that is to be used in the evaluation:

   ```
   Name =~'tr.*0'
   Name LIKE 'tr%0'
   ```

2. The following expressions evaluate to TRUE for all rows or resources that contain 1, 3, 5, 6, or 7 in the column or attribute that is called IntList, which is an array:

   ```
   IntList|<{1,3,5..7}
   IntList in (1,3,5..7)
   ```

3. The following expression combines the previous two so that all rows and resources that have a name beginning with 'tr' and ending with '0' and have 1, 3, 5, 6, or 7 in the IntList column or attribute will match:

   ```
   (Name LIKE "tr%0")&&(IntList|<(1,3,5..7))
   (Name=~'tr.*0') AND (IntList IN {1,3,5..7})
   ```

Chapter 3. Components Provided for Monitoring

The major components of Cluster Systems Management monitoring tool are the Resource Monitoring and Control (RMC) subsystem and certain resource managers. These are described in the following sections.

Resource Monitoring and Control Subsystem

The Resource Monitoring and Control (RMC) subsystem monitors and queries resources. The RMC daemon manages an RMC session and recovers from communications problems.

The RMC subsystem is used by its clients to monitor the state of system resources and to send commands to resource managers. The RMC subsystem acts as a broker between the client processes that use it and the resource manager processes that control resources.

Resource Managers

A resource manager is a process that maps resource and resource-class abstractions into calls and commands for one or more specific types of resources. A resource manager is a stand-alone daemon. The resource manager contains definitions of all resource classes that the resource manager supports. A resource class definition includes a description of all attributes, actions, and other characteristics of a resource class.

See the man pages for the RMC and ERRM commands or *Cluster Systems Management for Linux Technical Reference* to learn how to access the resource classes and manipulate their attributes through the command line interface.

The following resource managers are provided:

Audit Log resource manager (IBM.AuditRM)
Provides a system-wide facility for recording information about the system's operation, which is particularly useful for tracking subsystems running in the background. (See "Using the Audit Log to Track Monitoring Activity" on page 9 and "Audit Log Resource Manager" on page 20 for details.)

Distributed Management Server Resource Manager (IBM.DMSRM)
Manages a set of nodes that are part of a system management cluster. This includes monitoring the status of the nodes and adding, removing, and changing attributes of the nodes in the cluster. (See "Distributed Management Server Resource Manager" on page 20 for details.)

Event Response resource manager (IBM.ERRM)
Provides the ability to take actions in response to conditions occurring on the system. (See "Event Response Resource Manager" on page 21 for details.)

File System resource manager (IBM.FSRM)
Monitors file systems. (See "File System Resource Manager" on page 23 for details.)

Host resource manager (IBM.HostRM)
Monitors resources related to an individual machine. The types of values that are provided relate to load (processes, paging space, and memory usage) and status of the operating system. It also monitors program activity from initiation until termination. (See "Host Resource Manager" on page 25 for details.)

Sensor resource manager (IBM.SensorRM)
Provides a means to create a single user-defined attribute to be monitored by the RMC subsystem. See "Sensor Resource Manager" on page 29 for details.

Audit Log Resource Manager

The Audit Log subsystem is implemented as a resource manager within the RMC subsystem. It has two resource classes, IBM.AuditLog for subsystem definitions and IBM.AuditLogTemplate for audit-log-template definitions. Entries in the audit log are called records. Records can be added, retrieved, and removed through actions on a specific subsystem or on the subsystem class. The template definition class contains a description of each record type that a subsystem can add to the audit log. The template definition contains the data type, a descriptive message, and other information for each subsystem-specific field within the record.

There are typically two types of clients for the audit-log subsystem, subsystems that need to add records to the audit log, and users who extract records from the audit log through the command line. The formatted message for each record provides a concise description of the situation and allows a user to easily see at a high level what has been happening on the system.

Audit Log Resource Class

Each resource of this class represents a subsystem that will be adding records to the audit log. A resource of this class must be added before the subsystem can add records to the audit log. The resource can be added as part of the installation of the subsystem or at runtime.

The following properties can be monitored for this resource class:

RecordsAdded
: Reflects the current number of records in the audit log. Whenever records are added to the audit log, this value is updated.

RecordsRemoved
: Conveys which records have been removed. The following data elements comprise the value of this attribute:

  RecordCount Reflects the total number of records in the audit log after the records identified by SeqNumRanges have been removed.

  SeqNumCount
  : Reflects the total number of elements in the SeqNumRanges array. The number of ranges in that array is actually SeqNumCount/2.

  SeqNumRanges
  : Each consecutive pair of CT_INT64 integers defines an inclusive range of sequence numbers of records that have been deleted.

AuditLogSize
: Reflects the amount of disk space in bytes that the audit log uses.

Audit Log Template Resource Class

This resource class holds all audit log templates. An audit log template describes the information that exists in each audit log record that is based on the template. In addition, an audit log template contains information on how to present records that use the template to an end user. Each template corresponds to a resource within this class. The attributes of this resource class are internal.

Distributed Management Server Resource Manager

The distributed management server resource manager (IBM.DMSRM) controls the managed node (IBM.ManagedNode) resource class and the node group (IBM.NodeGroup) resource class. The distributed management server resource manager runs on the node designated as the management server and is automatically started by the RMC subsystem.

Managed Node Resource Class

The program name of this resource class is IBM.ManagedNode. It runs on the management server and is started by the RMC subsystem. It is controlled by the distributed management server resource manager.

The following dynamic attributes can be monitored for the IBM.ManagedNode resource class:

ConfigChanged
When a persistent attribute value changes, this attribute is asserted.

PowerStatus Monitors the power status of the node. The valid states are OFF (0), ON (1), and UNKNOWN (127).

Status Represents the current accessibility status of the node. **Accessibility** is defined as the ability to successfully ping the node. The valid states are UNREACHABLE (0), REACHABLE (1), and UNKNOWN (127).

Predefined Conditions for Managed Node Resource Class

The following table shows the predefined conditions and example expressions that are available for the IBM.ManagedNode resource class.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Notes
NodeReachability	Status!=1	An event is generated when a node in the network cannot be reached from the management server.	Status=1	The event is rearmed when the node can be reached again.	None.
NodeChanged	ConfigChanged=1	An event is generated when a node definiton in the ManagedNode resource class changes.	None.	None.	NodeNames = {localnode}

Node Group Resource Class

The program name of the node group resource class is IBM.NodeGroup. The node group resource class runs on the management server.

The following dynamic attributes of the node group resource class can be monitored:

ConfigChanged
When a persistent attribute value changes, this attribute is asserted.

Event Response Resource Manager

The system administrator interacts with the Event Response resource manager (ERRM) through the ERRM command-line interface.

When an event occurs, ERRM runs user-configured commands, which can include scripts provided by RSCT. A command and its attributes are a type of action, and many actions can be configured for a single Event Response resource. An action consists of a name, a command to be run, and other variables. You specify the range of times when the command is run (day, start time, and end time). If the condition occurs at a time outside the specified time ranges, the command is not run, and if all of the actions within this Event Response resource have the same time ranges, none of the commands are run. If no time ranges are specified, the command is always run. There are also event and rearm event flags that specify the events for which the command is run. Three options are allowable; only event set, only rearm event set, or both flags set.

The Event Response Resource Manager (ERRM) is automatically started when the RMC subsystem is started.

Although performance is important, ensuring that no events are lost and that the user's commands are run is of greater importance. Other factors outside the control of ERRM may affect performance as well (for example, network load, system load, and the performance of other required subsystems).

The only user ID that can define, undefine, and modify ERRM resources is root. All other users have read access to ERRM resources. Security is governed by the RMC daemon, which authenticates clients and performs authorization checks. No security audits are generated, and no encryption mechanisms are used. ERRM communicates only with other local subsystems on the same node.

Information is handled as follows:

- Files that contain internal trace output that is useful to a software service organization in resolving problems are written to **/var/ct/IW/log/mc/IBM.ERRM/trace**.
- Core files are written to the **/var/ct/IW/run/mc/IBM.ERRM** directory.
- The Audit Log facility records events and the actions taken by ERRM in response to those events, such as changes in the registration of Conditions with RMC.

There are three **Event Response** resource classes:

1. **Condition**

   The Condition resource class contains the necessary information (event expression and rearm expression) for the ERRM to register with the RMC for event notifications that the administrator deems important. Conditions contain essential information such as the resource attributes of the resource to be monitored, the event expression, and the optional rearm expression.

   Configuration of ERRM begins with the definition of a set of Condition resources. A Condition resource is registered with the RMC subsystem when the Condition resource is used in the definition of an active Association resource.

   Notes:

   a. Registration with RMC is necessary for monitoring to run. Registration does not occur when a new Condition resource is defined, but rather when the resource is used in the definition of an active Association resource.

   b. While monitoring a Condition on multiple nodes, if the RMC session with any one node is lost, the Condition's monitor status will be ″monitored but in error.″

2. **Event Response**

   An Event Response resource is configured by defining one or more actions. Each action contains the name of the action, a command, and other fields within the action attribute. The Event Response resource runs any number of configured commands when an event with an active association occurs. When an event occurs, all of the actions associated with its Event Response resource are evaluated to determine whether they should be run.

   Predefined responses are available to use and to serve as templates for creating your own responses. For a description of predefined responses and how to use them, see "Predefined Responses" on page 29. Scripts for notification and logging of events and for broadcasting messages to logged-in user consoles are provided in *Cluster Systems Management for Linux Technical Reference*.

   Note: Commands are run in parallel.

   See "Getting Started with the Monitoring Application" on page 7 for specific task information on how to configure actions for Event Response resources and Event Response resources for Conditions.

3. **Association**

   The Association resource class joins the Condition resource class together with the Event Response resource class. It contains a flag that indicates whether the association between the condition and the

event response is active. Event Responses and Conditions are separate entities, but for monitoring to take place, they need to be associated. An event cannot occur unless at least one Event Response is associated with a Condition. You can configure one or more actions for an Event Response, and one or more Event Responses for a Condition.

See “Getting Started with the Monitoring Application” on page 7 for information on how to get started using the capabilities of the Event Response resource manager to monitor your system.

File System Resource Manager

The File System resource manager (FSRM) manages file systems. It can do the following:

- List all file systems within the system.
- List only the file systems that match certain criteria.
- Obtain the status of a file system (mounted or unmounted).
- Obtain the values of the attributes of the file system.
- Monitor the percentage of disk space used for the file system.
- Monitor the percentage of i-nodes used for the file system.

There is one File System resource manager (FSRM) on a node. It is started implicitly by the RMC subsystem and is run only when an attribute of an FSRM resource class is monitored (thus cutting down on performance overhead).

To enforce security, only root can start the FSRM resource manager (although it is strongly recommended that the FSRM resource manager not be started manually). Security is governed by the RMC daemon, which authenticates clients and performs authorization checks. No security audits are generated, and no encryption mechanisms are used. The FSRM communicates only with other local subsystems on the same node and with the RMC subsystem. The FSRM has no direct contact with clients.

Information is handled as follows:

- Files that contain internal trace output that is useful to a software service organization in resolving problems are written to **/var/ct/IW/log/mc/IBM.FSRM**.
- Core files are written to the **/var/ct/IW/run/mc/IBM.FSRM** directory.

These attributes of a file system resource can be monitored:

OpState Monitors whether the current file system operational state is online (mounted) or offline (unmounted).

PercentTotUsed
Represents the percentage of space that is used in a specific filesystem so that preventative action can be taken if the amount available is approaching a predefined threshold. For example, `/tmp PercentTotUsed`, `/var PercentTotUsed`.

PercentINodeUsed
Represents the percentage of i-nodes that are in use for a specific file system; for example, `/tmp PercentINodeUsed`.

Predefined Conditions for Monitoring File Systems

The following table shows the predefined conditions and examples of expressions that are used to monitor the file system:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
File system state	OpState != 1	An event is generated when any file system goes offline.	OpState == 1	The event is rearmed when any file system comes back online.	all	n/a
File system i-nodes used	PercentINodeUsed > 90	An event is generated when more than 90% of the total i-nodes in any file system are in use.	PercentINode Used < 85	The event is rearmed when the percentage of i-nodes used in the file system falls below 85%.	all	n/a
File system space used	PercentTotUsed > 90	An event is generated when more than 90% of the total space of any file system is in use.	PercentTotUsed < 85	The event is rearmed when the space used in the file system falls below 85%.	all	n/a
/tmp space used	PercentTotUsed > 90	An event is generated when more than 90% of the total space in the **/tmp** file system is in use.	PercentTotUsed < 85	The event is rearmed when the space used in the **/tmp** file system falls below 85%.	**/tmp**	n/a
/var space used	PercentTotUsed > 90	An event is generated when more than 90% of the total space in the **/var** file system is in use.	PercentTotUsed < 85	The event is rearmed when the space used in the **/var** file system falls below 85%.	**/var**	n/a
AnyNode FileSystem InodesUsed	PercentINodeUsed > 90	An event is generated when more than 90% of the total i-nodes in the file system are in use.	PercentINodeUsed < 75	The event is rearmed when the percentage of i-nodes used in the file system falls below 75%.	all	n/a
AnyNode FileSystem SpaceUsed	PercentTotUsed>90	An event is generated when more than 90% of the total space of the file system is in use.	PercentTotUsed <75	The event is rearmed when the percentage of space used in the file system falls below 75%.	all	n/a

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
AnyNodeTmp SpaceUsed	PercentTotUsed>90	An event is generated when more than 90% of the total space in the **/tmp** directory is in use.	PercentTotUsed <75	The event is rearmed when the percentage of space used in the **/tmp** directory falls below 75%	**/tmp**	Use **Name= '/tmp'** for select string.
AnyNodeVar Space Used	PercentTotUsed>90	An event is generated when more than 90% of the total space in the **/var** directory is in use.	PercentTotUsed <75	The event is rearmed when the percentage of space used in the **/var** directory falls below 75%	**/var**	Use **Name= '/tmp'** for select string.

Host Resource Manager

The Host resource manager allows system resources for an individual machine to be monitored, particularly resources related to operating system load and status.

The Host resource manager is started implicitly by the RMC subsystem only when an attribute of a Host resource class is first monitored (thus cutting down on performance overhead).

Security is governed by the RMC daemon, which authenticates clients and performs authorization checks. The Host resource manager runs as root. No security audits are generated, no encryption mechanisms are used, and there is no communication outside the node. The RMC daemon detects any unsuccessful authentication or authorization attempts. All interprocess communication is accomplished through pipes and shared memory.

Information is handled as follows:

- Files that contain internal trace output which is useful to a software service organization in resolving problems are written to **/var/ct/IW/log/mc/IBM.HostRM**.
- Core files are written to the **/var/ct/IW/run/mc/IBM.HostRM** directory.

The Host resource manager consumes minimal system resources during normal operation. This is because the following approaches have been implemented:

1. Memory, CPU, and other system resources are not consumed for attributes that are not monitored. If no attributes are monitored, the Host resource manager is not started.
2. To minimize disk access, information is maintained in memory as much as possible.
3. The sampling of attribute values is aligned as much as possible to minimize the sampling overhead, in particular, thread or process context swaps.

The Host resource manager has the following resource classes that you can use to monitor system resources:

Host (IBM.Host)

This resource class externalizes the attributes of a machine that is running a single copy of an operating system. Primarily the attributes included are those that are advantageous in predicting or indicating when corrective action needs to be taken. See “Host Resource Class” on page 26 for more details.

Program (IBM.Program)

This resource class allows a client to monitor attributes of a program that is running on a host. The program to monitor is identified by properties such as program name, arguments, etc. The resource class does not monitor processes as such because processes are very transient and therefore inefficient to monitor individually. See “Program Resource Class” on page 27 for more details.

Host Resource Class

The program name of this resource class is IBM.Host. It allows the following resources of a host system to be monitored:

1. Global state of active paging spaces (see “Monitoring the Global State of Active Paging Space”).
2. Total processor utilization across all active processors in the system (see “Monitoring Processor Utilization”).

Monitoring the Global State of Active Paging Space

The following attribute monitors the percentage of paging space in use:

PctTotalPgSpUsed

Represents the percentage of paging space in use for all active paging space devices in the system.

Predefined Conditions for Monitoring Global State of Active Paging Space

The following table shows the predefined condition that is available for monitoring paging space, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Paging percent space used	PctTotalPgSpUsed > 90	An event is generated when more than 90% of the total paging space is in use.	PctTotalPgSpUsed < 85	The event is rearmed when the percentage falls below 85%.

Monitoring Processor Utilization

The values represented for this attribute reflect total processor utilization across all of the active processors in a system.

This attribute can be monitored:

PctTotalTimeIdle

Represents the system-wide percentage of time that the processors are idle.

Predefined Conditions for Monitoring Processor Utilization

The following table shows the predefined condition that is available for monitoring system-wide processor idle time, and example expressions:

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description
Processor idle time	PctTotalTimeIdle>= 70	An event is generated when the average time all processors are idle at least 70% of the time.	PctTotalTimeIdle < 10	The event is rearmed when the idle time decreases below 10%.

Program Resource Class

The program name of this resource class is IBM.Program resource class. This resource class can monitor a set of processes that are running a specific program or command whose attributes match a filter criterion. The filter criterion includes the real or effective user name of the process, arguments that the process was started with, etc. The primary aspect of a program resource that can be monitored is the set of processes that meet the program definition. A client can be informed when processes with the properties that meet the program definition are initiated and when they are terminated. This resource class typically is used to detect when a required subsystem encounters a problem so that recovery actions can be performed and the administrator can be notified.

Program Definition

A program definition requires the program name and the user name of the owner of the program. The program should be identified by user name in addition to program name to avoid confusion when two or more programs have the same name. These attributes are defined as follows:

ProgramName
: Identifies the name of the command or program to be monitored. The program name is the base name of the file containing the program. This name is displayed by the **ps** command when the **–l** flag or **–o comm** is specified. Note that the program name displayed by **ps** when the **–f** flag or **–o args** is specified may not be the same as the base name of the file containing the program.

Filter
: Specifies a filter that selects a subset of all processes running the program identified by the attribute **ProgramName**. For example, the filter may limit the process set to those processes that are running **ProgramName** under the user name **foo**.

Note: Process IDs are not used to specify programs because they are transient and have no prior correlation with the program being run, nor can the restart of a program be detected because there is no way to anticipate the process ID that would be assigned to the restarted application.

For a process to match a program definition and thus be considered to be running the program, its name must match the ProgramName attribute value. In addition, the expression defined by the Filter attribute must evaluate to TRUE by using the properties of the process. The Filter attribute is a string that consists of the names of various properties of a process, comparison operators, and literal values. For example, a value of `user==greg` restricts the process set to those processes that run ProgramName under the user ID**greg**. The syntax for the Filter value is the same as for a string.

For more information on selection strings, see “Using Expressions” on page 11.

Processes must have a minimum duration (approximately 15 seconds) to be monitored by the IBM.Program resource class. (If a program runs for only a few seconds, all processes that run the program may not be detected.)

This attribute can be monitored: **Processes**

These elements of the **Processes** attribute can be monitored:

CurPidCount
: Represents the number of processes that currently match the program definition and thus are considered to be running the program.

PrevPidCount
: Represents the number of processes that matched the program definition at the last state change (previous value of **CurPidCount**).

CurrentList
: Contains a list of IDs for the processes that currently match the program definition and thus are considered to be running the program.

ChangeList
: Contains a list of IDs for the processes that were added to or removed from the **CurrentList** since the last state change. Whether the list represents additions or deletions

can be determined by comparing **CurPidCount** and **PrevPidCount**. If **CurPidCount** is greater, this list contains additions; otherwise, it contains deletions. Additions and deletions are not combined in the same state change.

For example, assume the six processes shown in the following **ps** output are running the **biod** program on node 1:

```
ps -e -o "ruser,pid,ppid,comm" | grep biod

root 7786 8040 biod

root 8040 5624 biod

root 8300 8040 biod

root 8558 8040 biod

root 8816 8040 biod

root 9074 8040 biod
```

To be informed when the number of processes running the specified program changes, you can define this event expression:

```
Processes.CurPidCount!=Processes.PrevPidCount
```

To be informed when no processes are running the specified program, you can define this event expression:

```
Processes.CurPidCount==0
```

Predefined Conditions for Monitoring Programs

This resource class is typically used to detect when a required subsystem encounters a problem so that some recovery action can be performed or an administrator can be notified. The following table shows the predefined conditions and examples of expression that are available for monitoring programs.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
sendmail daemon state	Processes .CurPidCount <=0	An event is generated whenever the **sendmail** daemon is not running.	Processes .CurPidCount> 1	The event is rearmed when the **sendmail** daemon is running.	**sendmail**	n/a
inetd daemon state	Processes .CurPidCount <=0	An event is generated whenever the **inetd** daemon is not running.	Processes .CurPidCount> 1	The event is rearmed when the **inetd** daemon is running.	**inetd**	n/a
MgmtSvrCfd Status	Processes .CurPidCount <=0	An event is generated when the **cfengine** daemon stops running.	Processes .CurPidCount> 1	The event is rearmed when the **cfengine** daemon starts running again.	CSM Mgmt Server	Use **ProgramName= 'cfd'** for the select string.

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
AnyNodeCfd Status	Processes .CurPidCount <=0	An event is generated when the **cfengine** daemon stops running.	Processes .CurPidCount> 1	The event is rearmed when the **cfengine** daemon starts running again.	all nodes	Use **ProgramName= 'cfd'** for the select string

Sensor Resource Manager

The Sensor resource manager makes the output of a user-written script known to the RMC subsystem as a dynamic attribute of a sensor resource. The Sensor resource manager determines when this attribute is run according to a specified interval. Thus, an administrator can set up a user-defined sensor to monitor an attribute of interest and then create expressions that contain Conditions and Responses with associated actions that are performed when the attribute has a certain value. For example, a script can be written to return the number of users logged on to the system. Then an ERRM Condition and Response can be defined to run an action when the number of users logged on exceeds a certain threshold.

Sensor Resource Class

The Sensor resource manager has one class, IBM.Sensor. Each resource in the IBM.Sensor resource class represents one sensor and includes information such as the script command, the user name under which the command is run, and how often it should be run. The output of the script causes a dynamic attribute within the resource to be set. This attribute can then be monitored in the typical way.

See the **mksensor** man page for details on how to set up a sensor.

Predefined Condition for Sensor Resource Class

The following table shows the predefined condition and example expression that is available for the IBM.Sensor resource class.

Condition Name	Event Expression	Event Description	Notes
CFMRootModTimeChanged	″String!=\@P″	An event is generated when a file under /cfmroot is modified, added, or deleted.	Selection String = 'Name=″CFMRootModTime″'

Predefined Responses

The following predefined responses are shipped as templates or as starting points for monitoring.

See “Using Expressions” on page 11 for a summary of the data types and operators that you can use in selection strings for a customized response.

Response Name	Command
BroadcastEventsAnyTime	/usr/sbin/rsct/bin/wallevent
CForce	/opt/csm/bin/cforce -a
EmailEventsToRootAnyTime	/usr/sbin/rsct/bin/notifyevent root
DisplayEventsAnyTime	/usr/sbin/rsct/bin/displayevent admindesktop:0
LogEventsAnyTime	/usr/sbin/rsct/bin/logevent /var/log/csm/systemEvents

Response Name	Command
MsgEventsToRootAnytime	/usr/sbin/rsct/bin/msgevent root

Predefined Commands, Scripts, Utilities, and Files

You can use the following commands, scripts, utilities, and files to control Monitoring on your system. See the command man pages or *Cluster Systems Management for Linux Technical Reference* for detailed usage information.

ERRM commands

chcondition Changes any of the attributes of a defined condition.

lscondition Lists information about one or more conditions.

mkcondition Creates a new condition definition which can be monitored.

rmcondition Removes a condition.

chresponse Adds or deletes the actions of a response or renames a response.

lsresponse Lists information about one or more responses.

mkresponse Creates a new response definition with one action.

rmresponse Removes a response.

lscondresp Lists information about a condition and its linked responses, if any.

mkcondresp Creates a link between a condition and one or more responses.

predefined-condresp
Creates or resets the default monitoring conditions and responses.

rmcondresp Deletes a link between a condition and one or more responses.

startcondresp
Starts monitoring a condition that has one or more linked responses.

stopcondresp Stops monitoring a condition that has one or more linked responses.

RMC Commands

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

lsactdef Lists (displays) action definitions of a resource or resource class.

lsrsrc Lists (displays) resources or a resource class.

lsrsrcdef Lists a resource or resource class definition.

mkrsrc Defines a new resource.

refrsrc Refreshes the resources within the specified resource class.

rmrsrc Removes a defined resource.

Scripts and Utilities

ctsnap Gathers configuration, log, and trace information for the Reliable Scalable Cluster Technology (RSCT) product.

displayevent Notifies the specified user of an event by displaying it on the X-Window at the terminal of the user.

logevent Logs event information generated by the Event Response resource manager to a specified log file.

lsaudrec Lists records from the audit log.

msgevent Sends a message to the specified user.

notifyevent Emails event information generated by the Event Response resource manager to a specified user ID.

rmaudrec Removes records from the audit log.

rmcctrl Manages the Resource Monitoring and Control (RMC) subsystem.

wallevent Broadcasts an event or a rearm event to all users who are logged in.

Files

Resource Data Input File
Defines resources and attribute values of a resource or resource class.

rmccli General Information File
Contains information global to the RMC command line interface.

Chapter 4. Diagnostic Information

Files are created in the **/var/ct/IW/log/mc/***Resource Manager* directory to contain internal trace output that is useful to a software service organization for resolving problems. An internal trace utility tracks the activity of the resource manager daemon. Multiple levels of detail may be available for diagnosing problems. Some minimal level of tracing is on at all times. Full tracing can be activated with the command:

```
traceson -s IBM.HostRM
```

Minimal tracing can be activated with the command:

```
tracesoff -s IBM.HostRM
```

where **IBM.HostRM** is used as an example of a resource manager.

Resource Manager Diagnostic Files

All trace files are written by the trace utility to the **/var/ct/IW/log/mc/***Resource Manager* directory. Each file in this directory that is named **trace<.***n***>** corresponds to a separate run of the resource manager. The latest file that corresponds to the current run of the resource manager is called **trace**. Trace files from earlier runs have a suffix of *.n*, where *n* starts at 0 and increases for older runs.

Use the **rpttr** command to view these files. Records can be viewed as they are added for an active process by adding the **–f** option to the **rpttr** command.

Any core files that result from a program error are written by the trace utility to the **/var/ct/IW/run/mc/***Resource Manager* directory. Like the trace files, older core files have a *.n* suffix that increases with age. Core files and trace files with the same suffix correspond to the same run instance.

The **log** and **run** directories have a default limit of 10MB. The resource managers ensure that the total amount of disk space used is less than this limit. Trace files without corresponding core files are removed first when the resource manager is over the limit. Then pairs of core and trace files are removed, starting with the oldest. At least one pair of core and trace files is always retained.

Recovering from RMC and Resource Manager Problems

This section describes the tools that you can use to recover from infrastructure problems. It tells you how to determine if the components of the monitoring system are running and what to do if the RMC subsystem or one of the resource managers should abnormally stop. Common troubleshooting problems and solutions are also described.

The Audit Log, Event Response, File System, and Host resource managers recover from most errors because they have few dependencies. In some cases, the recovery consists of terminating and restarting the appropriate daemon. These resource managers can recover from at least the following errors:

1. Losing connection to the RMC daemon, probably caused by the terminating of the RMC daemon or another system problem.
2. Programming errors that cause the process to abnormally terminate. In this case, the SRC subsystem restarts the daemon. This includes errors such as incorrect memory references and memory leaks.
3. The **/var** or **/tmp** directories filling up. When this happens, core and trace files cannot be captured.

In addition, all parameters received from the RMC subsystem are verified to avoid impacting other clients that may be using the same resource manager.

The following tools are described:

1. **ctsnap** command

2. SRC-controlled commands
3. **rmcctrl** command for the RMC subsystem
4. Audit log

ctsnap Command

For debugging purposes, the **ctsnap** command can be used to **tar** the RSCT and resource-manager programs and send them to the software service organization. The **ctsnap** command gathers system configuration information and compresses the information into a **tar** file, which can then be downloaded to disk or tape and transmitted to a remote system. The information gathered with the **ctsnap** command may be required to identify and resolve system problems. See the man page for the **ctsnap** command for more information.

SRC-Controlled Commands

The RMC subsystem and the resource managers are controlled by the System Resource Controller (SRC). They can be viewed and manipulated by SRC commands. For example:

To see the status of all resource managers, type:

```
lssrc -g rsct_rm
```

To see the status of an individual resource manager, type:

```
lssrc -s rmname
```

where *rmname* can be:

- IBM.AuditRM
- IBM.DMSRM
- IBM.ERRM
- IBM.FSRM
- IBM.HostRM
- IBM.Sensor

To see the status of all SRC-controlled subsystems on the local machine, type:

```
lssrc -a
```

To see the status of a particular subsystem, for example, the RMC subsystem, which is known to SRC as **ctrmc**, type:

```
lssrc -s ctrmc
```

The SRC has these commands:

- **lssrc**
- **startsrc**
- **stopsrc**
- **traceson**
- **tracesoff**

For more information, see the command man pages.

For more information about SRC, see *System Management Concepts: Operating System and Devices*, which is available at http://www.ibm.com/servers/aix/library.

Recovery Support for RMC Using rmcctrl

The RMC command **rmcctrl** controls the operation of the RMC subsystem and the RSCT resource managers. It is not normally run from the command line, but it can be used in some diagnostic environments; for example, it can be used to add, start, stop, or delete an RMC subsystem. For more information, see the **rmcctrl** command man page or *Cluster Systems Management for Linux Technical Reference*.

Tracking ERRM Events with the Audit Log

The audit log is a system-wide facility for recording information about the system's operation. It can include information about the normal operation of the system as well as system problems and errors. It is meant to augment error log functionality by conveying the relationship of the error relative to other system activities. All detailed information about system problems is still written to the operating system error log.

Records are created in the audit log by subsystems that have been instrumented to do so. For example, the Event Response subsystem runs in the background to monitor conditions defined by the administrator 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 Event Response subsystem activity as well as that of other subsystems through the **lsaudrec** command.

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 (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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.

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
Department LJEB/P905
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following trademarks apply to this book:

IBM and AIX are registered trademarks of International Business Machines Corporation.

Linux is a registered trademark of Linus Torvalds.

Red Hat and RPM are trademarks of Red Hat, Inc.

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

Other company, product, and service names may be the trademarks or service marks of others.

Publicly Available Software

Cluster Systems Management for Linux includes software that is publicly available:

cfengine	A software package that is licensed under GPL and is used to create customization scripts.
Conserver	An application that adds logging and multi-user access for remote administration of serial ports, using locally installed multi-port serial interfaces and/or ″reverse-telnet″ to console servers.
DBD-CSV, DBI	Licensed by GPL or Artistic, these are dynamically loaded Perl modules.
fping	Licensed by BSD, this is executed as a separate binary.

Perl	Practical Extraction and Report Language is licensed under the Artistic license.
Pidentd	Public domain program by Peter Eriksson that implements the RFC-1413 identification server.
SQL-Statement	Licensed under GPL or Artistic, this is a dynamically loaded Perl module.

This book discusses the use of these products only as they apply specifically to the Cluster Systems Management for Linux product.

Note: The distribution for these products includes the source code and associated documentation. All copyright notices in the documentation must be respected. You can find version and distribution information for each of these products that are part of your selected install options in the **README** file.

Index

R

S

T

U

V

Readers' Comments — We'd Like to Hear from You

IBM Cluster Systems Management for Linux®
Monitoring HOWTO
Version 1 Release 1

Publication No. SA22-7852-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7852-00

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie NY 12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

SA22-7852-00

Cut or Fold
Along Line

Program Number: 5799-GNJ

SA22-7852-00