IBM Cluster Systems Management for Linux®

Administration Guide

Version 1.1

SA22-7873-01

IBM Cluster Systems Management for Linux®

Administration Guide

Version 1.1

SA22-7873-01

Note!
Before using this information and the product it supports, read the information in “Notices” on page 77.

Third Edition (December 2001)

This edition of the *IBM Cluster Systems Management for Linux Administration Guide* applies to IBM Cluster Systems Management for Linux Version 1.1, program number 5765–E88, 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.

Contents

About This Book

Chapter 1 of this book is the updated version of the *IBM Cluster Systems Management for Linux Overview HOWTO* (SA22-7857-00), and Chapters 2-5 are the updated version of the *IBM Cluster Systems Management for Linux Monitoring HOWTO* (SA22-7852-00) that were last published in June, 2001. Chapter 6 is the updated version of the *DCEM Administration Guide*. This book describes IBM Cluster Systems Management for Linux (CSM) and how to use the CSM monitoring function. It also provides reference material about the RSCT Monitoring application, including a summary of components, messages, and predefined monitoring functions.

Who Should Use This Book

This book is intended for system administrators who want to use IBM Cluster Systems Management for Linux. The system administrator should have experience in UNIX® administration and networked systems.

How to Use This Book

This book contains an overview of IBM Cluster Systems Management for Linux (CSM) and information on how to use the Monitoring application, which is available through the command line interface (CLI). This book also contains reference information that describes the underlying system, diagnostic information, commands and scripts, and tables of predefined conditions, expressions, and responses that are packaged with the application.

Highlighting

The following highlighting conventions are used in this book:

Bold	Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.
Italic	Identifies parameters whose actual names or values are to be supplied by the user.
`monospace`	Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.

ISO 9000

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

Related Information

The following references contain more information about IBM Cluster Systems Management for Linux:

- *IBM Cluster Systems Management for Linux Planning and Installation Guide*
- *IBM Cluster Systems Management for Linux Remote Control Guide and Reference*
- *IBM Cluster Systems Management for Linux Technical Reference*

How to Obtain Publications

The IBM 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:

The file names are as follows:

- *IBM Cluster Systems Management for Linux Administration Guide*, am7LXadm.pdf
- *IBM Cluster Systems Management for Linux Planning and Installation Guide*, am7LXstp.pdf
- *IBM Cluster Systems Management for Linux Remote Control Guide and Reference*, am7LXrem.pdf
- *IBM Cluster Systems Management for Linux Technical Reference*, am7LXcmd.pdf

Publications for IBM 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).

How to Reach Us by E-Mail

If you would like to contact us by e-mail, send your comments to cluster@us.ibm.com.

Chapter 1. Cluster Systems Management Overview

IBM Cluster Systems Management for Linux (CSM) provides a distributed system management solution for machines, or *nodes*, that are running the Linux operating system. With this software, an administrator can easily set up and maintain a Linux cluster by using functions like monitoring, hardware control, and configuration file management. The concepts and software are derived from IBM Parallel System Support Programs for AIX® (PSSP) and from applications available as open source tools.

Within a CSM cluster, nodes can be added, removed, changed, or listed (with persistent configuration information displayed about each node in the list). Commands can be run across nodes or node groups in the cluster, and responses can be gathered. Nodes and applications can be monitored as to whether they are up or down; CPU, memory, and system utilization can be monitored; and automated responses can be run when events occur in the cluster. Configuration File Manager is provided for synchronization of files across multiple nodes. A single management server is the control point for the CSM cluster.

Note that CSM manages a loose cluster of machines. It does not provide high availability services or fail-over technology, although high-availability clusters can be part of the set of machines that CSM is managing. For information on IBM high availability solutions, see http://www.ibm.com/servers/eserver/pseries/solutions/ha/index.html.

The following diagram (Figure 1) shows a remote control compatible hardware and networking configuration for CSM with IBM @server Cluster 1300 (xSeries™) 330 nodes. (See the *IBM Cluster Systems Management for Linux Remote Control Guide and Reference* for additional hardware configuration diagrams.) The Management Server connects to the Management VLAN and the Cluster VLAN through Ethernet adapters. The terminal server, an Equinox Serial Provider (ESP) in this example, connects to the Management VLAN through its Ethernet adapter, and to the cluster nodes through their serial (COM) ports as shown. (An ESP-16 can connect up to 16 nodes. Other terminal servers may have different capacities.) The nodes must be connected to the Cluster VLAN through their Ethernet adapters, and directly or indirectly to an IBM Remote Supervisor adapter (RSA) through their ISP ports. The Management VLAN connects to the RSAs in select nodes. (One RSA is required for every 10 nodes.) The RSAs connect to their own node's ISP port, and up to 9 more node ISP ports are daisy-chained from there. Configuration for a Public VLAN is optional and can be defined by the system administrator.

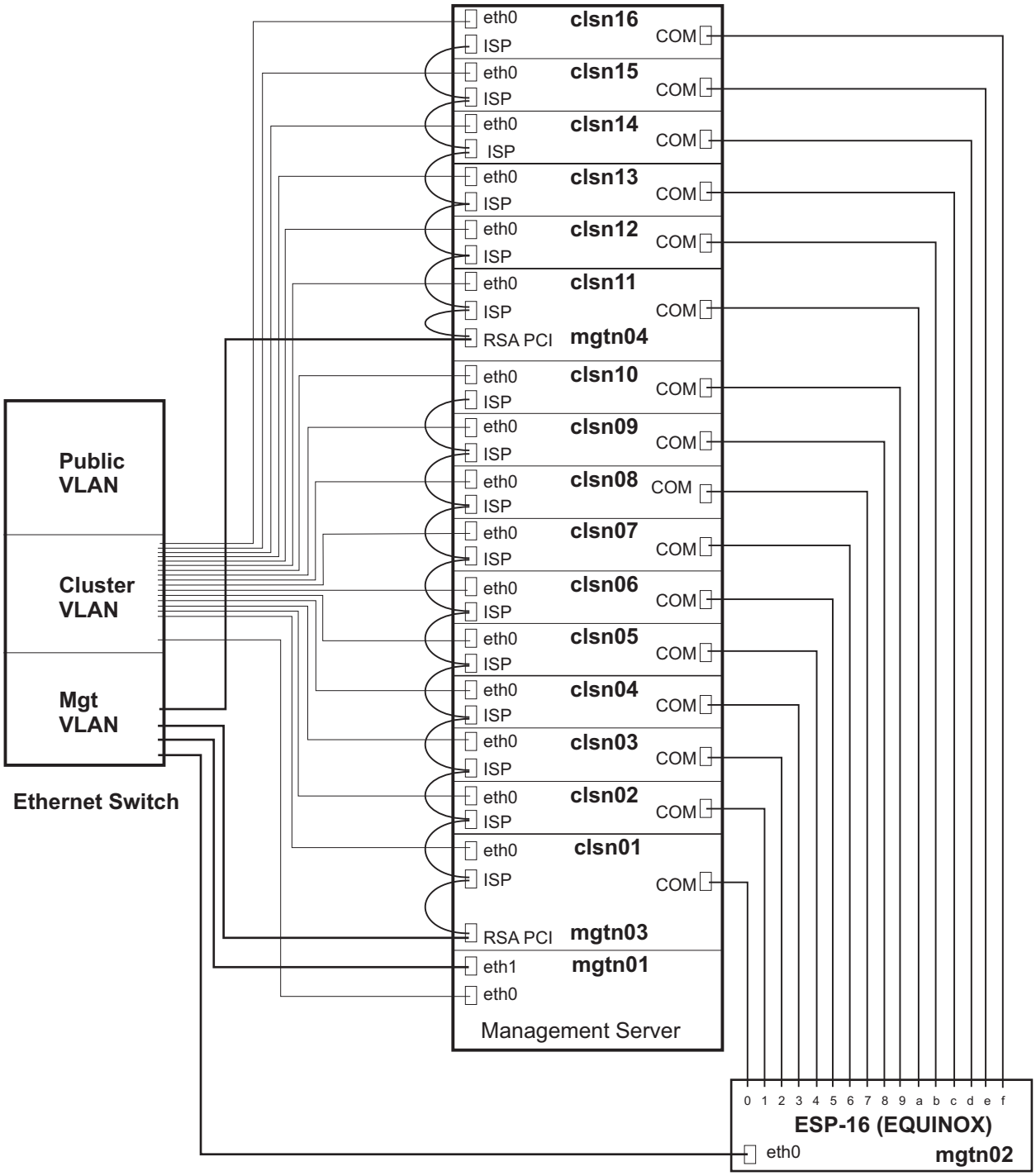


Figure 1. CSM Remote Control Hardware and Networking Configuration for IBM xSeries 330 Nodes

More information is provided on CSM tasks as follows:

1. Setting up Cluster Systems Management
2. Managing node and node group information
3. Monitoring and controlling hardware
4. Running remote commands on multiple nodes
5. Monitoring system events
6. Managing and synchronizing configuration files for all nodes
7. Providing security
8. Diagnosing problems

Setting Up Cluster Systems Management

The *IBM Cluster Systems Management for Linux Planning and Installation Guide* provides a simple process for installing and configuring CSM on an existing Linux system, or for doing a full installation of CSM and Linux. The installation process allows you to do the following:

1. Install IBM RSCT and CSM code on the management server.
2. Automatically add nodes to the cluster during the installation process.
3. Install and configure IBM RSCT and Cluster Systems Management on all nodes in a cluster from a single management server.

For more information, see the man pages or *IBM Cluster Systems Management for Linux Technical Reference* for the following set up commands and files:

installms Installs the management server.

addnode Can be used instead of **definenode** and **installnode** for suitable installations.

definenode Gathers all the information necessary to install the nodes.

installnode Installs the nodes and brings up the necessary servers on them.

nodedef **file** Node definition file for cluster nodes.

getmacs Automatically gathers and stores MAC addresses in the CSM database.

kscfg.tmpl Linux operating system configuration file generated by Kickstart.

monitorinstall Displays the status of the installation on each of the nodes.

setupks Sets up the management server to install Kickstart on the nodes.

Managing Node and Node Group Information

The distributed management server provides a set of commands for managing nodes. It stores information about nodes in a central repository, and it defines static and dynamic node groups. These definitions are then accessible to the Configuration File Manager (**cfm**) command for configuration file management, the **dsh** command for running shell commands remotely, for hardware control, and for monitoring the cluster by using the Event Response subsystem (ERRM). All of these functions make use of the definitions stored by the node group and node repository. Thus, a node group is defined in only one place and is then accessible for use by other functions.

Persistent information on each node can be kept, including operating system type, hostname, machine type, model and serial number. Currently, the user must enter this information manually to store on each node. In the future, the cluster infrastructure will collect much of this information automatically when a node joins the cluster. In addition, the status of the node is determined periodically by means of the **fping** command.

The node and node group commands are built on top of a Perl DBI layer backed by a set of DBDs (database drivers) so that data can be stored in a variety of formats and shared with other tools.

See the man pages or the *IBM Cluster Systems Management for Linux Technical Reference* for details on the following commands that manage node and node group information:

addnode Adds and installs a node to the CSM domain.

chnode Changes an attribute of a node in the CSM cluster database.

dmsctrl Allows **fping** and power status parameters to be changed.

lsnode Displays information about the nodes in the CSM cluster, for example, the cached status on whether the node has been reachable.

nodegrp Defines node groups within the CSM cluster for use by other functions such as the configuration file manager, the **dsh** command, the event response subsystem, and the hardware control commands.

rmnode Removes a node from the CSM cluster database.

whichdb Specifies the repository for node information.

Monitoring and Controlling Hardware

You can control the hardware on remote nodes by using the remote control commands. For example, you can control computers on a ship from an office on the mainland, provided the correct connectivity exists.

See the man pages or *IBM Cluster Systems Management for Linux Technical Reference* for details on the following commands:

1. **rconsole**

   Opens a remote console.
2. **rpower**

   Boots and resets hardware, powers hardware on and off, and queries the power state.

See the *IBM Cluster Systems Management for Linux Remote Control Guide and Reference* for additional information on controlling hardware.

Running Commands Remotely

The distributed shell (**dsh**) command runs commands remotely across multiple nodes. It optionally can use any underlying remote shell that is specified by the user; for example, a remote secure shell that complies with the IETF (Internet Engineering Task Force) Secure Shell protocol. By default, **rsh** is used.

The **dsh** command can retrieve a complete list of the nodes in the CSM cluster or the list of nodes in a specified node group.

See the man pages or *IBM Cluster Systems Management for Linux Technical Reference* for details on the following commands:

dsh Issues remote shell commands and the options associated with them to multiple nodes.

dshbak Presents formatted output from the **dsh** command.

Configuration File Manager

Configuration File Manager provides a file repository for the common configuration files among nodes in a cluster. In general, all the configuration files that need to be shared are stored in one location on the management server. Changes to these files are propagated and synchronized throughout the cluster. Though the files are common, there are mechanisms to allow for variations based on groups, IP address, and host name.

Configuration File Manager is built on top of the GNU software package cfengine. The cfengine software package is a scripting package that uses a class-based decision structure to test and configure UNIX-like systems attached to a TCP/IP network. There are many capabilities built into cfengine itself, which a system administrator can use over and above what Configuration File Manager uses.

Configuration File Manager greatly enhances the copy functionality and usability of cfengine by providing the concept of a repository. Instead of requiring an administrator to write a cfengine script to keep files up to date, the repository allows automatic updating without script changes.

Note: CFM is based on cfengine, which currently uses a host list file (**/etc/opt/csm/cfd.conf**) to determine which hosts can access the configuration files that it controls. This may be a security vulnerability due to IP-address spoofing or a compromised DNS. This potential security vulnerability will be addressed in the next release.

See the man pages or *IBM Cluster Systems Management for Linux Technical Reference* for details on the **cfm** and **cforce** commands.

At the time this document was written, detailed information on cfengine could be found at the following URL: **http://www.cfengine.org**.

Monitoring System Events

A flexible distributed system monitoring application is provided by CSM. This monitoring application allows the administrator to define conditions of interest to monitor on a system. An event occurs when a monitored condition of interest reaches a threshold that is defined in an event expression. When an event occurs, automated responses to the event take place. Multiple actions can be defined as components of a response, including notification, running a predefined script, or running a user-defined script.

A set of commands is provided for setting up the monitoring application to meet your needs. A set of predefined conditions and responses is provided to be used as is or to be copied and modified. System resources that can be monitored include:

- File systems
- Programs
- System resources
- Node availability
- Other resources by means of sensors

The monitoring application, its components, and predefined conditions and responses are described in detail later in this book. Command syntax, descriptions, and examples are available as integrated man pages or in the *IBM Cluster Systems Management for Linux Technical Reference*.

Security Considerations

Security is provided by the operating system – only root can run functions or modify data. Flexibility in the degree of security required by a specific environment is provided by remote shells that conform to the IETF (Internet Engineering Task Force) Secure Shell protocol. Remote shells can be specified using the **dsh** command. Network security for other functions is built on the **identd** function.

See “Authorization” for details on authorization, and the **dsh** man page or the *IBM Cluster Systems Management for Linux Technical Reference* for details on how to specify the remote shell of your choice by using the DSH_REMOTE_CMD environment variable.

dsh Security

The distributed shell **dsh** command uses the underlying **rsh** security protocol, or any underlying remote shell that is specified by the user, such as a remote secure shell that complies with the IETF Secure Shell protocol. By default, **rsh** is used. It is the system administrator’s responsibility to configure and enable remote shell access to other systems, and to fulfill the particular security obligations of a specified environment. For more information, see the man pages or *IBM Cluster Systems Management for Linux Technical Reference* for details on the **dsh** and **dshbak** commands.

Authorization

CSM implements authorization using the access control list (ACL) file. You can create a new ACL file to apply access control to resource classes. Or you can use the default ACL file, which contains the following 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 an 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, query attribute values, and 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.

Authentication

CSM implements authentication (identity verification) 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 also be downloaded from one of the following URLs:

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

identd must 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
```

RMC Remote Security

The ACL file on the management server should look similar to Example 1 in "Examples of ACL File Stanzas". The ACL file on a managed node should look similar to Example 2 in "Examples of ACL File Stanzas".

ACL File Stanza Syntax

A stanza begins with a line containing the stanza name, which must start in column one. A stanza line consists of leading white space (one or more blanks, tabs, or both) 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.

Examples of ACL File Stanzas

1. The ACL file on the management server should look similar to the following example.

```
IBM.PreManagedNode
  root@clsn01.pok.ibm.com        *  rw
  clsn01.pok.ibm.com             *  r
  root@LOCALHOST                 *  rw  # root on this node always has access
  LOCALHOST                      *  r   # Everyone else on this node can only read

IBM.ManagedNode
  root@clsn01.pok.ibm.com        *  rw
  clsn01.pok.ibm.com             *  r
  root@LOCALHOST                 *  rw  # root on this node always has access
  LOCALHOST                      *  r   # Everyone else on this node can only read

IBM.NodeGroup
  root@clsno1.pok.ibm.com        *  rw  # root on this node always has access
  clsn01.pok.ibm.com             *  r   # Everyone else on this node can only read
  root@LOCALHOST                 *  rw  # root on this node always has access
  LOCALHOST                      *  r   # Everyone else on this node can only read
```

2. The ACL file on the managed node should look similar to the following example.

```
IBM.ManagementServer
  root@clsn01.pok.ibm.com  * rw # Grant root on c175n13.ppd.pok.ibm.com r/w access
  clsn01.pok.ibm.com       * r  # Everyone else on c175n13.ppd.pok.ibm.com has read-only access
  root@LOCALHOST            * rw # root on this node always has access
  LOCALHOST                * r  # Everyone else on this node has read-only access

OTHER
    clsn01.pok.ibm.com     * r  # The default denies write access to everyone from c175n13.ppd.pok.ibm.com
    root@LOCALHOST              * rw # root on this node always has access
    LOCALHOST                  * r  # Everyone else has read-only access
```

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

1. For resource class Class_A:

```
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
```

   - user1 at sys1 has permission to read and write all resource instances.
   - 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.
   - 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.
   - root on the machine containing the ACL file can read and write both the resource class and all of its resource 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.

2. For Class_B:

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

- 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.

3. For all other resource classes (represented by OTHER):

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

- root at **sys1** has permission to read both the resource class and all of its resource instances.
- root on the machine containing the ACL file can read and write both the resource class and all of its resource instances.

How to 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**.

Chapter 2. Monitoring for Linux

The 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 file system 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 email 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.

See “Chapter 3. Using the Monitoring Application” on page 17 for more details.

CSM uses RMC to monitor the system and to perform many of its operations. For information about the command line interface to the RMC subsystem, see *IBM Cluster Systems Management for Linux Technical Reference*. For information on RMC diagnostic information, see “Recovering from RMC and Resource Manager Problems” on page 46. For authorization and modifying the ACL file, see “Security Considerations” on page 6.

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

To understand and use conditions, you need to know about the following:

- Resource class
- Monitored resource
- Monitored attribute
- Event expression
- Rearm expression
- Sensors

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 22 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, you have to 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

Condition Component	Description	Example
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. The Monitoring application allows you to 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.

To create a new response you have 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

If you want to define different responses based on when the event occurs, you can associate multiple responses with a condition. For example, you might have a work day response and a weekend response, each containing one or more actions. 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

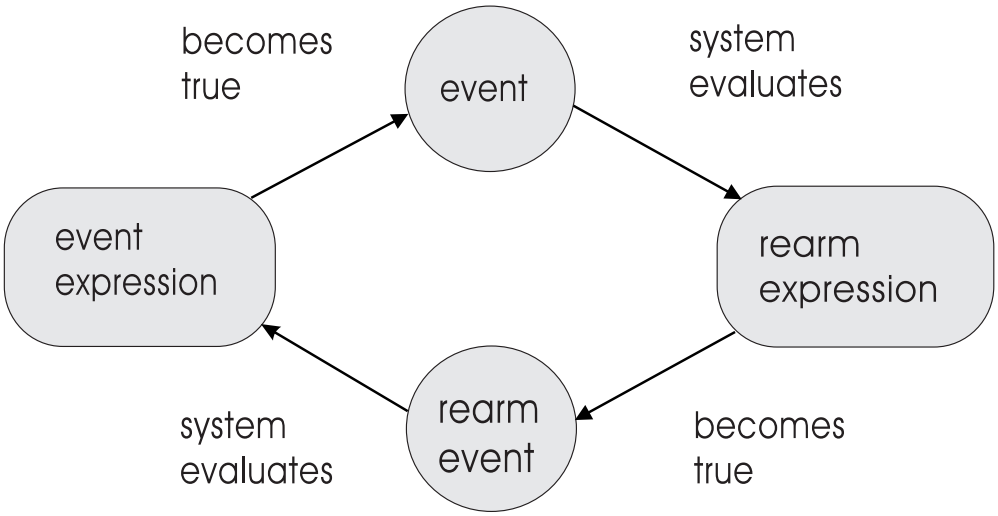
After monitoring for the condition begins, the system evaluates the event expression to see if it is true. When the event expression becomes true, an *event* occurs that automatically notifies all of the associated event responses, which causes each event response to run its defined actions.

The event expression and the rearm expression work together as follows when a condition is monitored:

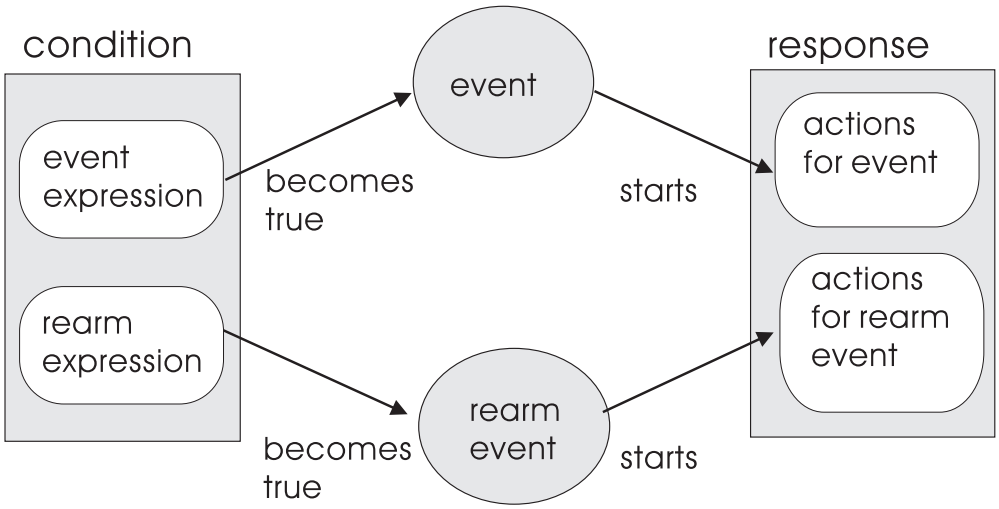
1. First, the event expression is evaluated.
2. When the event expression becomes true, an event occurs, and the specified actions are taken.
3. The system begins evaluating the rearm expression.
4. When the rearm expression becomes true, the *rearm event* occurs, which automatically starts the actions defined for the rearm event.
5. When the rearm event occurs, the system returns to evaluating the event expression.

Note: 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.

This cycle is illustrated below:



The interactions are illustrated below:



Chapter 3. Using the Monitoring Application

This chapter describes planning for monitoring your system and tracking system events, and using and modifying the predefined scripts, expressions, commands, and responses packaged with the Monitoring application. Predefined components and how to use them are described in detail in “Chapter 4. Components Provided for Monitoring” on page 31.

Planning What to Monitor in Your System

First, you 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 use it as a template to create a customized condition, or create a completely new 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 optionally, the rearm event.

Planning How to Respond to Detected Conditions

A set of predefined responses comes installed with your system (see “Predefined Responses” on page 43). Each response has one or more actions associated with it. Each action can be configured to fit your particular work environment and schedule.

The predefined actions are:

- Sending e-mail 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 with the **write** command (see the **msgevent** command man page).

You can also write your own commands that correct or mitigate conditions and run them using 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.

Getting Started with the Monitoring Application

This section describes how to start using the Monitoring application’s command line interface to:

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

See the man pages or the *IBM Cluster Systems Management for Linux Technical Reference* for detailed usage information.

How to Associate a Response with a Condition

To associate a response with a condition, use the following commands:

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 open the audit log. To log events to a user-defined file, use the **logevent** script.

How to Stop Monitoring

To stop monitoring for a condition, use the **stopcondresp** command.

Monitoring from the Command Line

The following examples show how to monitor your system using the command line interface. See the man pages or the *IBM Cluster Systems Management for Linux Technical Reference* for detailed usage information.

This example shows the sequence of setting up a new condition and response to have an event and rearm events respond accordingly:

1. Use the **mkcondition** command to create a condition you want to monitor.
2. Use the **mkresponse** command to create a response for the condition. One response can include multiple actions.
3. Use the **startcondresp** command to associate the condition with the response and start monitoring for the condition.
4. When the condition's event expression becomes true, an event occurs.
5. The actions defined in the response associated with the condition are examined to see which ones should be invoked. The action commands are then run in parallel.
6. No further events will occur for this condition until the rearm expression becomes true.
7. When the condition's rearm expression becomes true, a rearm event occurs.
8. The actions defined in the response associated with the condition are examined to see which ones should be invoked. The action commands are then run in parallel.
9. No further events will occur for this condition until the event expression becomes true.
10. Steps 4 - 9 then repeat.
11. Use the **stopcondresp** command to stop monitoring for the condition.

The following examples show uses of individual commands and sample command output:

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 associate a condition with a response without starting monitoring, use the **mkcondresp** command. For example, to associate the condition ″FileSystem space used″ with the responses ″Broadcast event on-shift″ and ″E-mail root anytime″, type:

```
mkcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root anytime"
```

5. 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"
```

6. 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"
```

7. 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"
```

8. To view events or the actions taken in response to the events, type:

```
lsaudrec -l
```

9. To define a response with the name ″E-mail root anytime″ that has an action named ″E-mail root″, to be used anytime Saturday and Sunday and uses the command **/usr/sbin/rsct/bin/notifyevent root** for both events and rearm events, type:

```
mkresponse -n "E-mail root" -d 1+7 \
-s "/usr/sbin/rsct/bin/notifyevent root" -e b "E-mail root anytime"
```

For a complete list of predefined commands, scripts, and utilities, see “Commands, Scripts, Utilities, and Files” on page 43.

Tracking Monitoring Activity

For information about monitoring events, rearm events, actions, and errors that have occurred, view the audit log using the **lsaudrec** command. See the **lsaudrec** man page and “Using the Audit Log to Track Monitoring Activity” on page 20 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 *IBM Cluster Systems Management for Linux Technical Reference*.

Using Scripts

The *IBM 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.

displayevent Displays an event or a rearm event to a specified X-window display.

logevent Appends a formatted string containing the specifics of an event to a user-specified text file.

msgevent Sends an event or rearm event to a specified user's console.

notifyevent Captures event information and sends it using UNIX mail to a specified user ID.

wallevent Broadcasts a message to all users who are logged in.

For complete descriptions of these scripts, see *IBM 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, 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 35 for a detailed description of ERRM):

- **ERRM_ATTR_NAME** - The display name of the dynamic attribute used in the expression that caused this event to occur.
- **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 24 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, operators, and operator order of precedence are described below. 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 23, 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
~	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

Operator	Description	Left Data Types	Right Data Types	Example	Notes
== =	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″ results in 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
>< 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

Operator	Description	Left Data Types	Right Data Types	Example	Notes
|| 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

Operators	Description
<	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 4. Components Provided for Monitoring

The major components of the 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. These resource classes are accessible and their properties can be manipulated by the user through the command line.

See the man pages for the commands or the *IBM 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 20 and "Audit Log Resource Manager" on page 32 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 "Managing Node and Node Group Information" on page 3 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 35 for details.)

File System resource manager (IBM.FSRM)
Monitors file systems. (See "File System Resource Manager" on page 36 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 38 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 42 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 resource-class persistent attributes can be modified:

HostResponseTimeout
The time interval that the **fping** command waits to receive a response to a ping packet that is sent to a node.

FpingInterval The time interval between two invocations of the **fping** command.

The following resource persistent attributes can be retrieved:

ConsoleMethod	Console Method.
ConsolePortNum	Console port number. For ESP, this must be a single hexadecimal digit within the range 0 - f.
ConsoleServerName	Console server host name; for example, mgtn02.pok.ibm.com.
ConsoleServerNumber	Console server number; for example, some number, 1 - N. For ESP, this is the ESP number associated with the Equinox ESP system.
Hostname	Host name of the node.
HWControlPoint	Host name of the ASM PCI adapter to which the internal service processor (ISP) of this node is connected; for example, mgtn03.
HW Model	Hardware model of the node.
HWSerialNum	Hardware serial number of the node.
HWType	Hardware type of the node.
InstallDisk	Disk to be used for node installation.
InstallDiskType	Type of the disk to be used for node installation.
InstallMethod	Installation Method (csmonly or kickstart).
LParID	Logical partition ID (applicable to the Power PC platform only).
Macaddr	MAC Address.
OSDistribution	Operation System distribution.
OSKernel	Operating System kernel level.
OSType	Operating system type.
OSVersion	Operating system version.
PowerMethod	Determines the program to invoke for a specific type of hardware power control; for example, **netfinity** (which corresponds to **/opt/csm/bin/netfinity_power**).
SvcProcName	Internal service processor name; for example, node01.
UniversalId	Universal ID (This is the node identifier.)

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

ConfigChanged
Indicates that there are changes to one or more persistent resource class attributes.

ResourceDefined
Indicates that a new ManagedNode resource was created.

ResourceUndefined
Indicates that a ManagedNode resource was deleted.

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

ConfigChanged
Indicates that there are changes to one or more persistent resource attributes.

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).

ResourceDefined
Indicates that a new ManagedNode resource was created.

ResourceUndefined
Indicates that a ManagedNode resource was deleted.

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 persistent attributes of resources belonging to the node group resource class can be retrieved:

Name Specifies the node group name.

ValidateNodes
Indicates whether the nodes in the node group need to be checked to verify that they are part of the managed-node cluster.

MemberList Specifies a string array in which each element is the host name of a node.

SelectStr Specifies the select string to be applied to the nodes defined to the managed node resource class.

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

ConfigChanged
Indicates that there are changes to one or more persistent resource class attributes.

ResourceDefined
Indicates that a new NodeGroup resource was created.

ResourceUndefined
Indicates that a NodeGroup resource was deleted.

The following dynamic attribute of a node group resource can be monitored:

ConfigChanged
Monitors whenever a persistent resource class attribute changes.

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 a response, which can include one or more actions. An action consists of a name, a command to be run, and other information. 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.

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 43. Scripts for notification and logging of events and for broadcasting messages to logged-in user consoles are provided in the *IBM Cluster Systems Management for Linux Technical Reference*.

   Note: Commands are run in parallel.

   See “Getting Started with the Monitoring Application” on page 17 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 “Monitoring from the Command Line” on page 18 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 file system 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

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
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
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” 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 40 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. 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

Program definition 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 Optional. 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 22.

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

Condition Name	Event Expression	Event Description	Rearm Expression	Rearm Description	Monitored Resources	Notes
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.
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.

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
MsgEventsToRootAnytime	/usr/sbin/rsct/bin/msgevent root

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 *IBM 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 associated responses, if any.

mkcondresp Creates an association between a condition and one or more responses.

rmcondresp Deletes an association between a condition and one or more responses.

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

stopcondresp Stops monitoring a condition that has one or more associated 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.

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

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 5. Diagnostic Information

To diagnose problems, it is helpful to understand the relationship between CSM and the tools that it uses. These tools that CSM uses are described in the following table:

Tool	What It Does	CSM Interface
Perl DBI package	Stores database information in a variety of formats.	all ″**node**″ commands
Resource Monitoring and Control (RMC) subsystem	Monitors conditions and communicates with all nodes. RMC needs to be running on each node, and the security access control list (ACL) file needs to allow the nodes to communicate with the management server. See “Security Considerations” on page 6.	**ERRM** commands
dsh	Runs commands remotely on the nodes. Security needs to be set up on each node to allow this for the remote shell that is used by **dsh**. The default remote shell is **rsh**.	
fping	Periodically gets the status of each node.	**IBM.DMSRM**
cfengine	Transfers files for the Configuration File Manager.	**cforce**, **cquery** commands
pxelinux	Enables network booting during installation.	all ″**install**″ commands
atftp	Advanced tftp. Handles file transfers during installation.	all ″**install**″ commands

The following tips can help diagnose problems with a CSM cluster:

- To ensure that the database attributes are correct for each node, type:

  ```
  lsnode -Al
  ```

- To list the status of the RMC daemons, type:

  ```
  lssrc -a
  ```

- To review the audit log for monitoring events, type:

  ```
  lsaudrec
  ```

- If you have modified the RMC access control list (ACL) file, make sure that it is correct on each node. If the default permissions have been modified, the RMC ACL file is located at **/var/ct/cfg/ctrmc.acls**.
- Make sure the cfengine security file (**/etc/opt/csm/cfd.conf**) on the management server contains all the nodes.
- Examine the cfengine log file: **/var/log/cfengine.log**.
- If you are using the **rsh** as the remote shell for **dsh**, make sure that the **/root/.rhosts** file on each node contains the hostname of the management server.
- To test **dsh** access on all nodes, type:

  ```
  dsh -a date
  ```

See “Security Considerations” on page 6 for detailed information on authorization and the ACL file. See the ACL File FAQ in the *IBM Cluster Systems Management for Linux Planning and Installation Guide* for information on troubleshooting the RMC ACL file. The CSM Frequently Asked Questions Web page at http://www.ibm.com/servers/eserver/clusters/software/csm_faq.html can also be useful for diagnosing problems.

Resource Manager Diagnostic Files

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.

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.

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 *IBM Cluster Systems Management for Linux Technical Reference*.

Tracking ERRM Events with the Audit Log

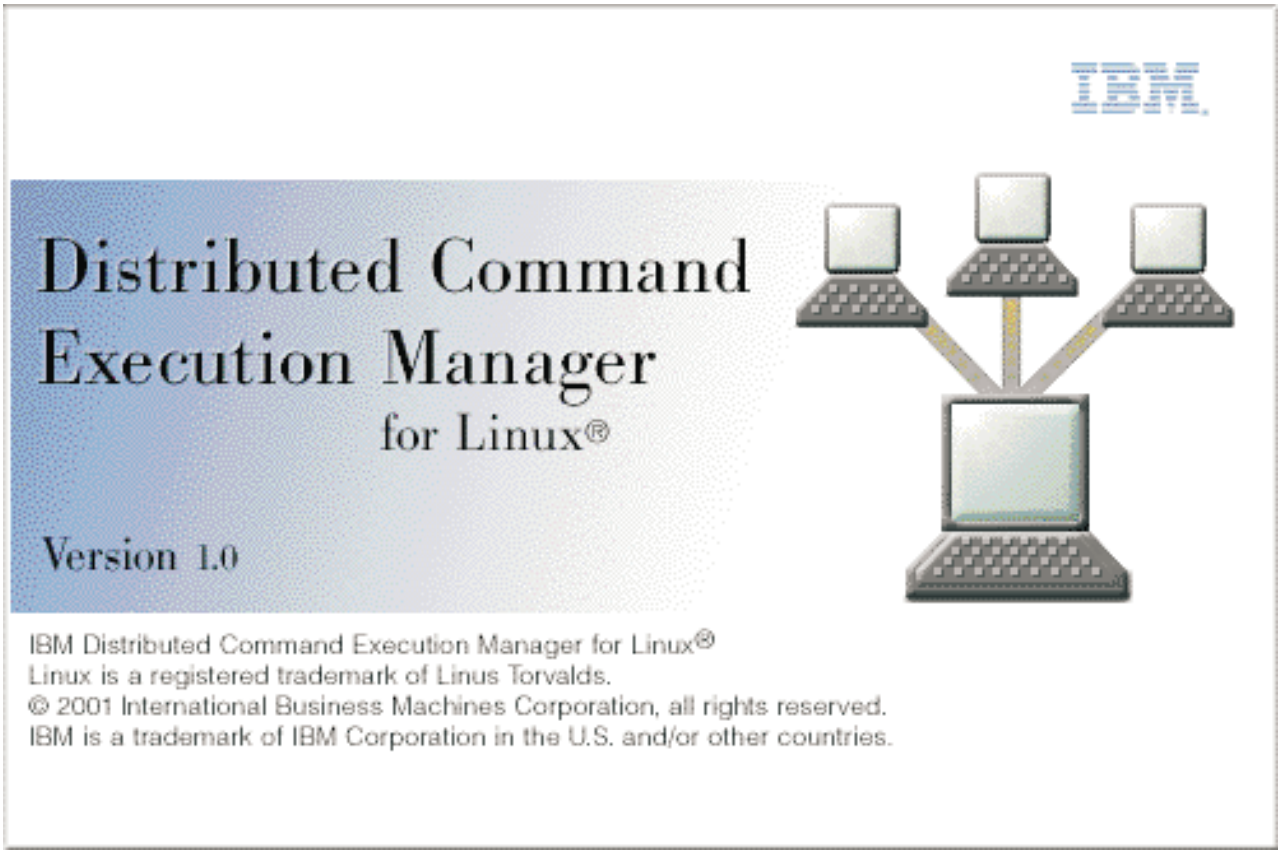
The audit log is a system-wide facility for recording information about the system's operation. It can include information about the normal operation of the system as well as system problems and errors. It is meant to augment error log functionality by conveying the relationship of the error relative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

Chapter 6. Distributed Command Execution Manager Overview

The Distributed Command Execution Manager (DCEM) provides a variety of services for a network of distributed machines. The DCEM graphical user interface allows you to construct command specifications for executing on multiple target machines, providing real-time status as commands are executed. You can enter the command definition, run-time options, and selected hosts and groups for a command specification, and you have the option of saving this command specification to use in the future. When you save a command specification, a Perl script is generated that you can run directly from a Linux or AIX command line. You can create and modify groups of hosts to use as targets for a command directly from DCEM. You can specify these groups by supplying host names for the group or by using dynamic queries on specific host attributes in a domain. DCEM also creates a log of all distributed command activity.

The DCEM startup window is shown in the following illustration.



Supported Platforms

You can use DCEM to manage systems running on the following operating systems:

- AIX 5L™ on PowerPC®
- Red Hat Linux

Setting Up DCEM

DCEM uses IBM Cluster Systems Management (CSM). CSM uses the Resource Monitoring and Control (RMC) subsystem. Installing DCEM requires that you install and correctly configure both CSM and RMC. In particular, DCEM uses the CSM **dsh** command to run commands on the nodes. For **dsh** to work, you must set up security on each node.

If you choose the default remote shell **rsh** , you must add the management server host name to the /.rhosts file on the nodes that will be managed. To make the managed hosts visible to DCEM (through the Browse Hosts dialog), you must create node definitions for each of the nodes to be managed in the CSM database on the management server where DCEM will run. You can create these definitions when you install CSM. However, if these node definitions do not exist, you can create them by using the following CSM **createnode** command:

```
/opt/csm/bin/createnode -M <host name>
```

See the *IBM Cluster Systems Management for Linux Technical Reference* for more information on the **createnode** command.

Starting DCEM

To start DCEM, type the following at the command line:

```
/opt/csm/dcem/bin/dcem
```

The DCEM command line options are as follows:

```
/opt/csm/dcem/bin/dcem [-h | --help] [-V | --version] [-v | --verbose] [-N | --groups
<group,group,group,...>] [-n | --hosts <host_name,host_name,host_name,...>]
[command_specification_name]
```

Command Syntax

Using the **dcem** command without options displays the Distributed Command Execution Manager dialog. From this dialog, you can create a new command specification or select from a list of saved command specifications.

Using the **dcem** command with the `command_specification_name` option causes DCEM to initialize the input fields in the main window with specified command data. The `command_specification_name` refers to the name used to save a command specification in the DCEM dialog. To send the command defined in the command specification to the specified hosts or groups, click the **Run** button in the dialog. The Execution Progress dialog shows the progress of the executed commands. To reset DCEM to the default values, click the **Defaults** button in the dialog.

You can also use the following flags:

- **-h | --help** writes the usage message for the **dcem** command to standard output.
- **-V | --version** writes version information to standard output.
- **-v | --verbose** runs the **dcem** command in debug mode and writes the command's verbose messages to standard output.
- **-N | --groups *group,group,group,...*** specifies the groups displayed in the Groups of hosts field of the DCEM dialog at startup. If you use this flag with a command specification name, the host names and groups that are stored as part of the command specification are ignored.
- **-n | --hosts *host_name,host_name,host_name,...*** specifies the hosts displayed in the **Host names** field of the DCEM dialog at startup. If you use this flag with a command specification name, the host names and groups that are stored as part of the command specification are ignored.

Example:

The following example specifies hosts and groups together with the `command_specification_name` parameter on the command line. Assume the **myCommand** command specification was saved with host names ***h1, h2, h3***, and groups of hosts ***g1, g2, g3***.

1. To run DCEM, type:

   ```
   dcem
   ```

2. To initialize the input fields with specified command specification name and groups, type:

   ```
   dcem --groups g4,g5 myCommand
   ```

   This results in the following output in the following GUI fields:

   ```
   Host names:{empty}

   Groups of hosts: g4,g5
   ```

3. To initialize the input fields with specified command specification name, groups, and hosts, type:

   ```
   dcem --groups g4,g5 --hosts h4 myCommand
   ```

   This results in the following output in the following GUI fields:

   ```
   Host names: h4

   Groups of hosts: g4,g5
   ```

4. To display the version of DCEM that is running, type:

   ```
   dcem -V
   ```

Note: : When you run DCEM from a remote host, run the **xhost +** command on that host from the machine you are using. On the machine running DCEM, run the **export** command, as follows:

```
export DISPLAY=IP address of the machine you are using
```

Using Distributed Command Execution Manager

The following panels help you to define command specifications:

- General panel
- Options panel
- Groups panel
- Dynamic Groups panel

The General and Options panels provide an interface for creating new command specifications and modifying previously saved command specifications. The Groups and Dynamic Groups panels provide an interface for creating and modifying groups of host machines.

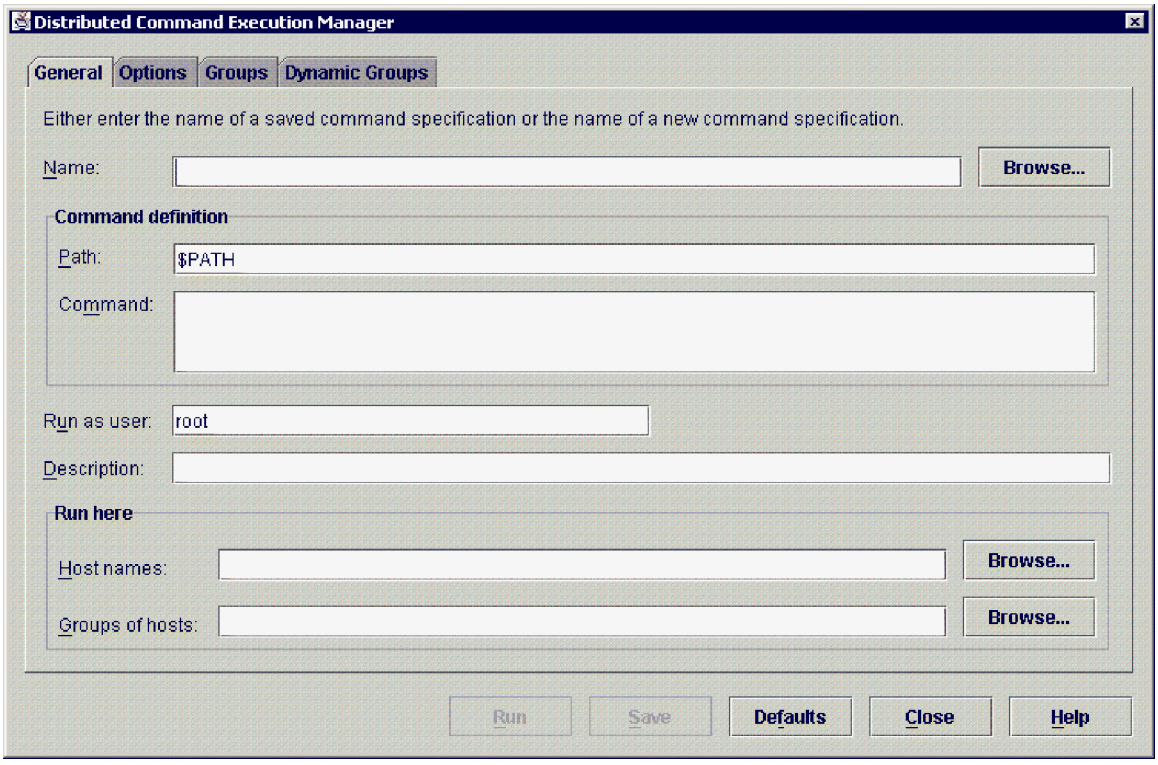
Creating Command Specifications

DCEM allows you to create, save, and edit command specifications, which reduces your time and effort when you repeatedly run the same command. A command specification consists of the following parts:

- The name of the command specification.
- A command definition including the path and command or inline script to run on remote host machines.
- The user name under which the command will run.
- A description of the command.
- A list of hosts or groups of hosts on which the command will run.
- Options for security, output streaming, and number of hosts on which to concurrently run the command.

General Panel

Use the following General panel to specify most of the information that is required to run commands on distributed hosts.



To create a command specification, you must provide, at a minimum, the following information in the text fields on this panel:

- **Name -** The name that identifies a command specification. When you create a new command specification, you must type a name for it in this field. The name field is not required for running a command specification, but is required when saving a specification.
- **Command -** The command or inline script to run, plus one or both of the following:
  - **Host Names -** The name of one or more hosts. You can type the name of any fully qualified host name as long as the host has a remote shell available, or type a list of fully qualified host names separated by commas or spaces. You can also select host names known to CSM from the Browse Hosts dialog, which displays when you click the **Browse** button next to the field.
  - **Groups of Hosts -** The name of one or more host groups. You can type a list of host groups separated by commas or spaces. You can also select host group names from the Browse Groups dialog, which displays when you click the **Browse** button next to the field. To use this selection dialog, you must first have created the groups of hosts. For more information, see Creating Groups of Hosts“Creating Groups of Hosts” on page 59.

The following fields, which contain default values, are also required, but are populated with default values:

- **Run as User -** The user name that the command will run under. By default, it is populated with the user name under which DCEM is running. You can edit this field. You must configure target machines to allow the user or machine under which DCEM is running to run as the user specified in this field. This configuration is specific to the remote shell used to run a command. (see Security Considerations and Remote Shells“Security Considerations and Remote Shells” on page 65)
- **Path -** The path that points to the actual location of the saved commands. The default value for this field is $PATH. You can edit this field.

  If you use the default $PATH in this field, the application does not delegate the local $PATH to the target hosts when the command is run. Instead, it prepends `export PATH=$PATH` to the command, where the $PATH variable referred to is the one set on the target machine.

To run commands found only in a specific directory, you can replace $PATH with that directory name. To guarantee that a directory is searched first, you can prepend the directory to $PATH, for example, /usr/bin:$PATH.

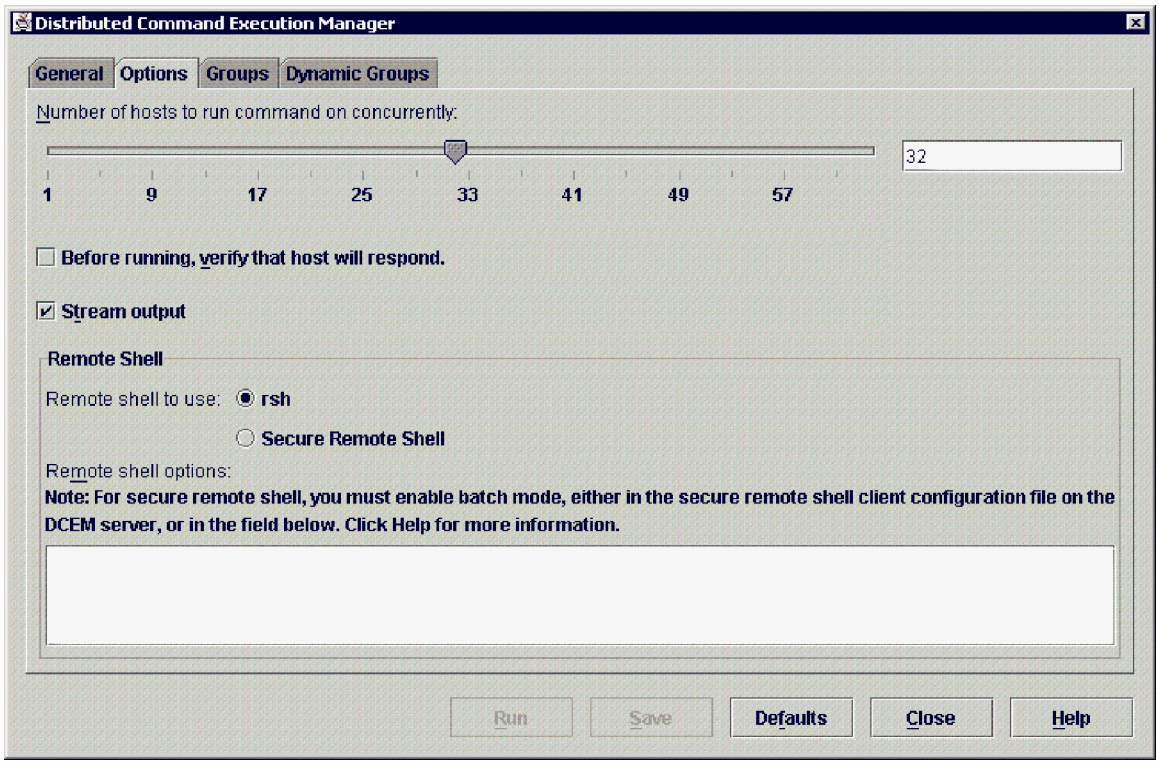
If you leave the **Path** field empty, it is the same as having $PATH.

The following field is optional and has no default:

- **Description -** An optional text description of the command specification. This description also displays in the Browse Command Specifications dialog to help you locate a particular saved command specification.

Options Panel

When you create a command specification, you can also change the options for executing the commands on the distributed hosts. Use the following Options panel to view or change options.



The Options panel displays the following options:

- **Number of hosts to run commands on concurrently -** The slider and text box display the number of hosts on which the command will run at the same time. You can specify between 1 and 64 hosts, with a default value of 32. To change this value, either drag the slider to a new value or specify a value in the text box.
- **Before running, verify that hosts will respond** - Select this check box if you want to determine whether the hosts are online and responding before you run a command specification. By default, this check box is not selected, which allows commands to be sent to all hosts without checking whether they are available.
- **Stream Output -** Select this check box if you want to display output in the Execution Progress dialog as it is received. When this box is not selected, the output is collected and displayed only after command execution completes.
- **Remote Shell -** Displays the remote shell under which distributed commands will be run. The remote shell options for **rsh** listed in this section are for the AIX platform. On Linux, these options could be different. The default shell on AIX is **rsh**, and if the secure remote shell is not installed, **rsh** will be the

only option available. You can enter options for either **rsh** or the secure remote shell in the text box provided. Enter the options as you would enter them on the command line.

Because the DCEM application is not interactive, you must configure the secure remote shell to run in batch mode. If the secure remote shell is not configured properly and you are prompted for a password during authentication, the command that you attempted to run cannot execute. You must then click the **Stop** button at the bottom of the Execution Progress dialog to stop the execution.

Options for **rsh** include the following:

– **-f** causes DCE credentials to be forwarded to remote hosts. This option is valid only if the underlying **rsh** uses Kerberos authentication and you have valid Kerberos credentials. It will be ignored if Kerberos 5 is not the current authentication method, and authentication will fail if the current DCE credentials are not marked forwardable.

– **-F** causes the credentials on the remote system to be marked forwardable (allowing them to be passed to another remote system). This setting will also be ignored if Kerberos 5 is not the current authentication method.

DCEM supports a variety of secure remote shells and has been tested, to a limited basis, using OpenSSH, a secure remote shell. If a secure remote shell supports batch mode, you must enable the batch mode. You can do this either in the secure remote shell client configuration file on the CSM server or in the secure remote shell options in the DCEM Options panel. (If the secure remote shell is installed, you can select the **Secure Remote Shell** radio button, and in the field where the options for this selection display, you can type the flag that enables batch mode.) For detailed information about secure remote shell client configuration, see the specific secure remote shell documentation.

Saving a Command Specification

When you have completed entering your command specification information, click the **Save** button to save it as a script. Command specifications that you save are stored as Perl scripts (see “Example of a saved command script” on page 65). The saved script contains all the information on the DCEM General and Options panels. Saved command specifications are located in the following directory:

home/dcem/scripts/*script file name*.pl

home is the home directory of the user under whose name the distributed command is run. *script file name*.pl is the name of a Perl script file containing a saved command specification.

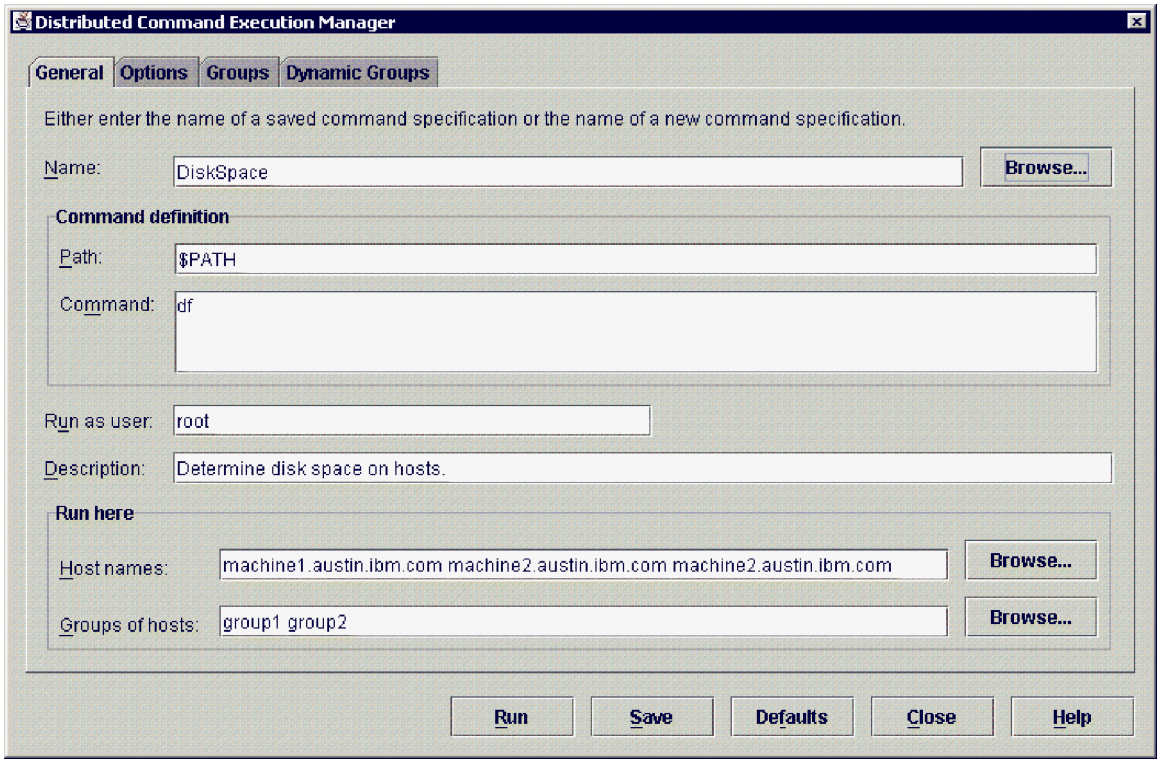
After you save a command specification, you can run it (see Running a Command on One or More Hosts“Running a Command on One or More Hosts”), view it, and select it from the Browse Command Specifications dialog.

Running a Command on One or More Hosts

You can run commands on multiple hosts using any of the following methods.

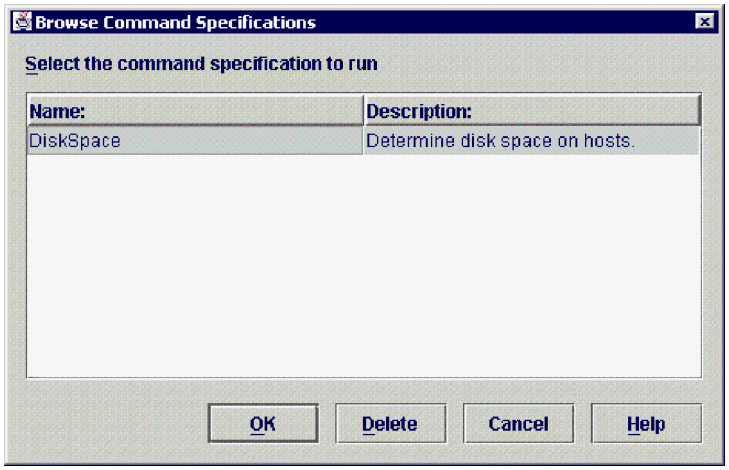
From the DCEM dialog

1. In the General panel, shown below, click the **Browse** button beside the **Name** text entry field.



This displays the Browse Command Specifications dialog.

2. In the Browse Command Specifications dialog, shown below, select a command specification from the list of existing command specifications, then click the **OK** button to load the selected command specification into the General and Options panels.



3. In the DCEM dialog, click the **Run** button to run the selected command specification on the specified hosts or groups of hosts.

From the DCEM dialog, you can also create a new command specification (see Creating Command Specifications“Creating Command Specifications” on page 51), then click the **Run** button to run the selected command specification on the specified hosts or groups of hosts.

Load the command specification from the command line

To load the command specification into the DCEM dialog directly from the command line, type the following:

```
/opt/csm/dcem/bin/dcem [<command_specification_name>]
```

Using the **dcem** command with the command_specification_name option causes DCEM to initialize the input fields in the General and Options panels of the DCEM dialog with specified command data. You can then click the **Run** button to send the command to the specified hosts.

Run the command specification script on the command line

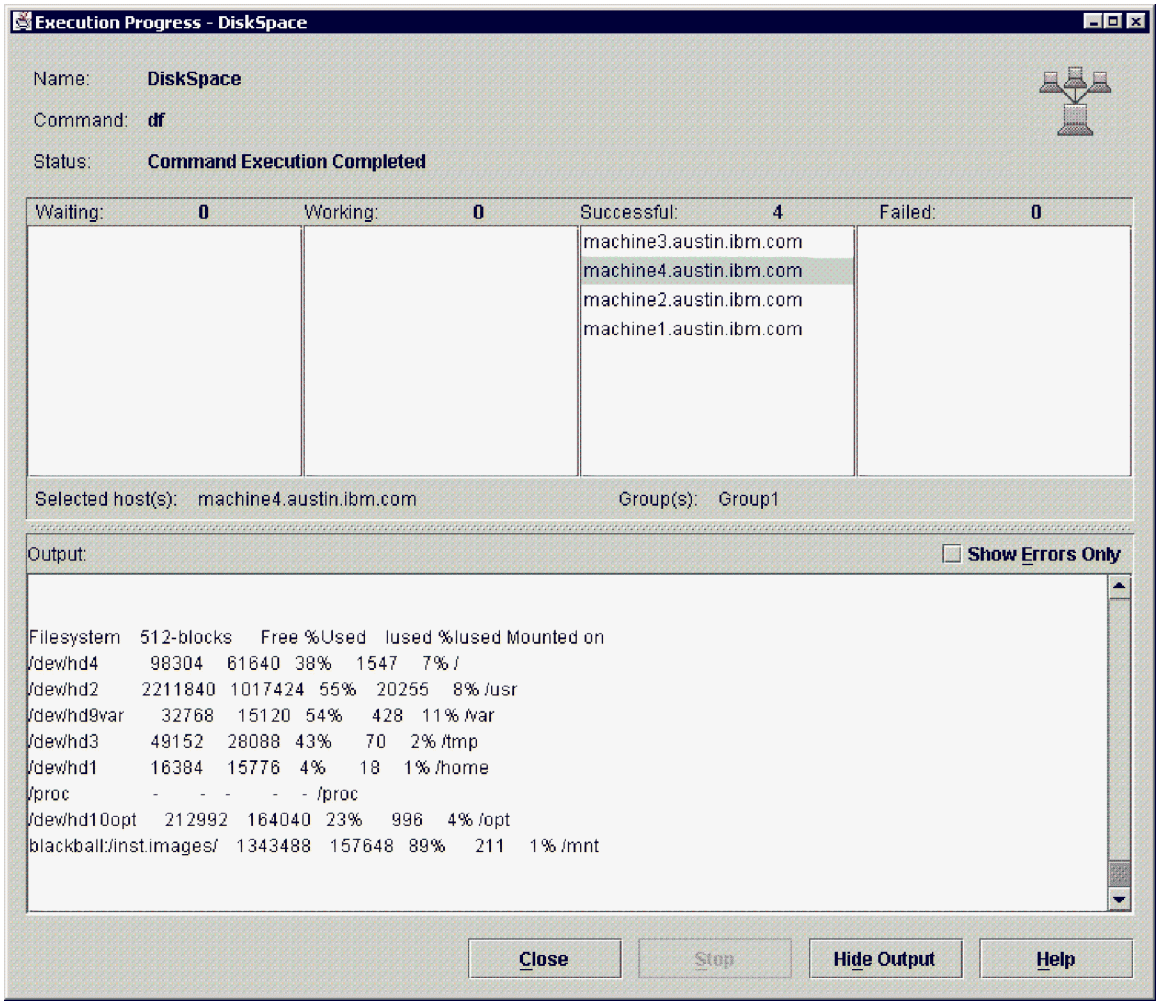
To run the command specification script directly on the command line, type the following:

```
<user_home_directory>/dcem/scripts/commandSpecificationName.pl [-debug] [-non_interactive]
[-format_output]
```

- **commandSpecificationName** - The name used to save a command specification using the DCEM dialog.
- **debug** - Verbose mode. Determines the actual execution string specified, for example, `/opt/csm/bin/dsh -f 11 -s -l root -N sysmgt-testbed "date"`.
- **non_interactive** - Does not prompt on the command line to run the command. This option is useful when invoking the command script from another script.
- **format_output** - Formats stdout output from all hosts. Output is grouped by host name.

Using the Execution Progress Dialog

After you click the **Run** button to run a valid command specification or command on valid hosts or groups of hosts, DCEM displays the following Execution Progress Dialog to show the status of the command execution on all of the hosts.



In this dialog, a series of lists show hosts on which command execution is in one of the following states:

- Waiting
- Working
- Successful
- Failed

The bottom of the window displays output from the command execution on selected hosts. To display output from a host, select its name from any of the lists. Multiple selections made from the Successful or Failed lists result in combined output in the output window. To view real-time output, select a host that is currently in the working state. The Waiting and Working lists do not support multiple selections.

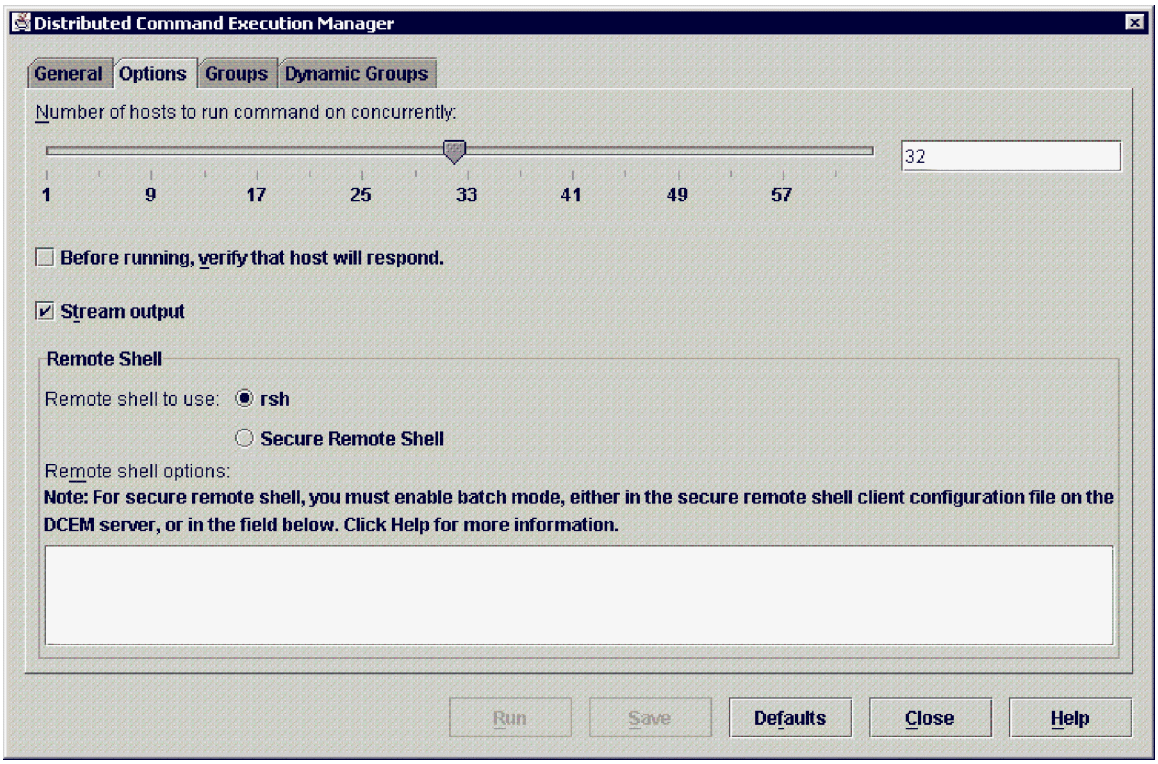
To stop the distributed command execution on all hosts, click the **Stop** button. If a command has not been completed on a host, command execution for that host is terminated. Hosts that have been stopped move to the Failed list. Command that have already completed may appear in either the Successful or Failed lists.

When the **Show Errors Only** check box is selected, the output area displays error messages for the selected hosts. **Stdout** messages are not displayed.

Selecting the **Close** button hides the Execution Progress Dialog, but does not stop execution on any hosts. If the DCEM dialog is closed before these hosts have completed execution, then these hosts will be stopped.

Fine-Tuning Run-Time Parameters

You can adjust several run-time options for your command using the DCEM Options panel, shown in the following illustration.



For example, if you are experiencing network problems and you want to improve performance, you can reduce the number of hosts on which the command specification will run at the same time. On the Options panel, the slider and the text box both display this value. The default is 32. Either of these can be modified with a new value between 1 and 64.

Note: When you change this value using the text field, the slider is only updated when the focus changes or the command is saved or run. It does not change as you type or when you press the **Enter** key.

You can also affect the amount of time it takes for commands to complete by selecting or deselecting the **Before running, verify that host will respond** check box. Selecting this check box allows you to invest the time to immediately check the host response. If there are problems, the wait time should be smaller than the minute typically taken for the remote shell command to time out.

You can change the default behavior of streaming the output (displaying it in the Execution Progress dialog as it is received) so that the output is collected and displayed for each host only after the command execution completes on that host and it is either in the successful or failed state.

You can also change the remote shell under which the commands run and specify options for that remote shell. The default remote shell is `rsh`.

Working with Groups of Hosts

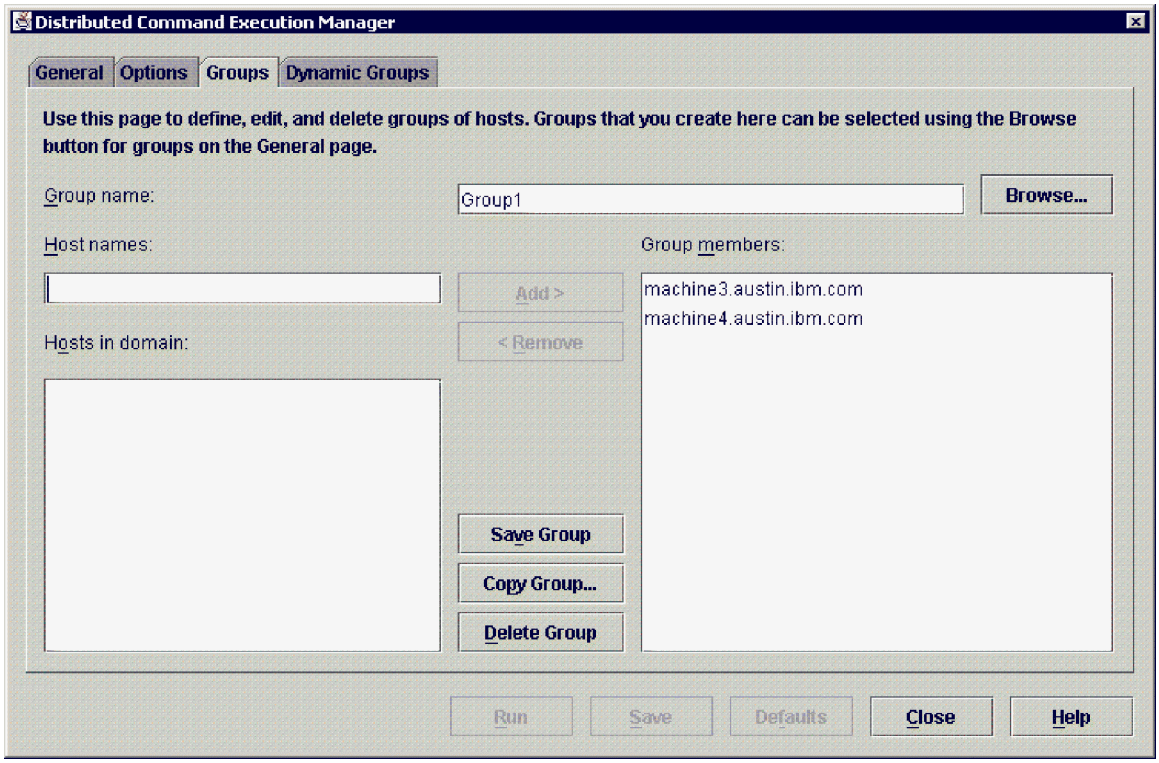
The ability to organize hosts into groups and to save these groups for later use can make it more convenient to run commands on the same groups of hosts repeatedly. Node groups can either be explicit lists of node host names, created by explicitly specifying each host, or they can be dynamic groups of hosts, created by specifying the desired selection criteria, such as ″Hostname like 'websrvr%'″. To create

explicit groups of hosts for use in its command specifications, use the Groups panel in the DCEM dialog. To create dynamic host groups to use in its command specifications, use the Dynamic Groups panel.

Note: By default, the root user has the authority to create groups and a non-root user cannot create groups unless special permission is set for that user. The access authority is defined in the `/var/ct/cfg/ctrmc.acls` file. You can modify this file, then run the **refresh -s ctrmc** command to refresh the systems. For more information, see the Security Considerations section of this book.

Groups panel

Use the Groups panel to create explicit groups of hosts. This panel provides an interface for editing, deleting, and copying the groups that you have created. The **Group members** list box contains a list of hosts that are already in the group.



Creating Groups of Hosts

Use the Groups panel to create a group of hosts. To create a group of hosts using the DCEM Groups panel, do the following:

1. Type the name of the group you are creating in the **Group name** field.
2. To add a host to the group, either type its name in the **Host names** field or select the host from the **Hosts in domain** list box, then click the **Add >** button. The hosts listed in the **Hosts in domain** list box are those hosts defined in CSM that have CSM client code installed.
3. To add hosts to the group that are not defined in CSM, type the host name in the **Host names** field, then click the **Add >** button.
4. To create the group of hosts, click the **Save Group** button.

Editing Groups of Hosts

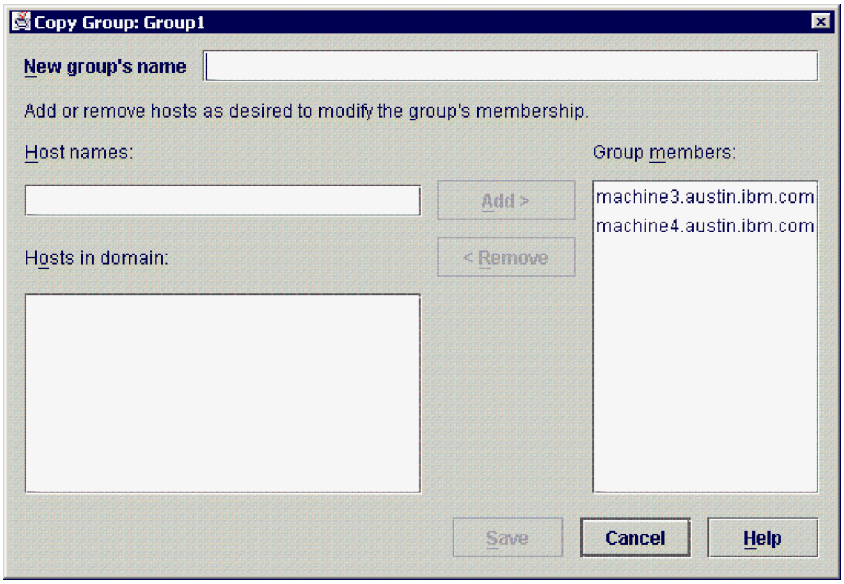
Use the Groups panel to edit an existing group of hosts. You can select group members to remove host names or add new host names. To edit a group of hosts, do the following:

1. In the **Groups** panel, click the **Browse** button beside the **Group name** box.
2. In the displayed **Browse Groups** dialog, select the group you want to edit.

3. To add a host to the selected group, either type a host name in the **Host names** field, or select a host from the **Hosts in domain** list box, then click the **Add >** button.
4. To delete a host from the selected group, select the hosts from the **Group members** list box, then click the **< Remove** button.
5. To save the changes, click the **Save Group** button.

Copying Groups of Hosts

Use the **Copy Group** dialog to copy an existing group of hosts. When you copy a group, you can also add new hosts to the group or remove hosts from the group before you save the copy.



To copy a group, do the following:

1. In the **Groups** panel, select the group you want to copy, then click the **Copy Group...** button.
2. In the **Copy Group** dialog, type the name of the new group in the **New group's name** field.
3. To add a host to the new group, either type a host name in the **Host names** field, or select a host from the **Hosts in domain** list box. To delete a host from the new group, select the host from the **Group members** list box, then click the **< Remove** button.
4. To save and copy the group, click the **Save** button.

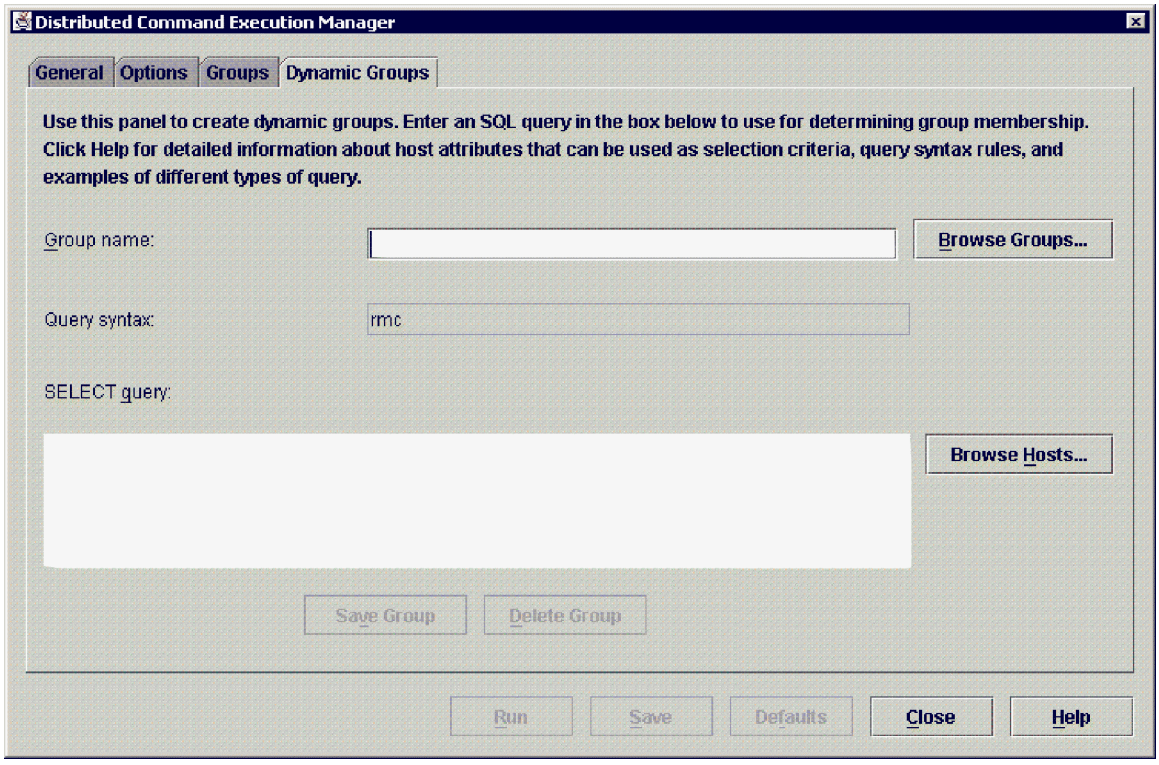
Deleting Groups of Hosts

Use the Groups panel to delete existing groups of hosts. To delete a group of hosts, do the following:

1. In the **Groups** panel, select the group you want to delete, then click the **Delete Group** button.
2. To confirm, click the **Delete** button in the **Deleting Host Groups** confirmation dialog. To cancel, click the **Cancel** button.

Dynamic Groups panel

The Dynamic Groups panel, shown in the following illustration, allows you to create dynamic groups of hosts based on a select (SQL-like) string. This select string is used to search the hosts database to dynamically determine the list of hosts in the group.



Creating a Dynamic Group of Hosts

To create a dynamic group of hosts using the DCEM Dynamic Groups panel, do the following:

1. Type the name of the group that you are creating in the **Group Name** field. Alphanumeric characters are allowed.
2. Type the select string in the **SELECT Query** field. This is the select string that is used to dynamically determine which nodes belong to this group. The syntax for this select string is determined by the type of node database you are working with (as indicated in the **Query syntax** box on this panel). If your node database is a **file** type, use normal SQL syntax. If it is of **rmc** type, use the syntax described in the Using Expressions section of this book.
3. To show the hosts that match your specified selection criteria after you specify a valid select string, click the **Browse Hosts** button.
4. To create the group of hosts, click the **Save Group** button.

Displaying Dynamic Group Host Members

To view host members of an existing dynamic group, do the following:

1. In the Dynamic Groups panel, click the **Browse Groups** button to display the **Browse Groups** dialog.
2. In the Browse Dynamic Groups dialog, select the dynamic group you want to view from the list of dynamic groups in the dialog. After you select the dynamic group, you are returned to the Dynamic Groups panel where the **Group name** field is filled in with the name of the selected group, and the **SELECT Query** field displays the chosen dynamic group's select string.
3. To view the hosts members of the selected dynamic group, click the **Browse Hosts** button beside the **SELECT Query** text field. The nodes that match the select string and that belong to the dynamic group are displayed.

   You can also use this feature to view hosts that satisfy the **SELECT Query** before actually creating a group. To do this, type the select string you want to use, then click on the **Browse Hosts** button. To view hosts matching a select string, you do not need to provide a group name.

Editing a Dynamic Group

Just as you cannot create a dynamic group by explicitly adding hosts to the group, you cannot edit a dynamic group by explicitly adding or removing hosts from the group. The only characteristic you can change about a dynamic group is its select string.

To edit an existing dynamic group, do the following:

1. In the Dynamic Groups panel, click the **Browse Groups** button to display the **Browse Groups** dialog.
2. In the Browse Groups dialog, select the dynamic group you want to edit.
3. In the **SELECT Query** field modify query. You can check the hosts members that will belong to the group, as defined by your select string, by clicking on the **Browse Hosts** button.
4. To save the selected dynamic group with its edited select string, click on the **Save Group** button.

Copying a Dynamic Group

To copy a dynamic group, do the following:

1. In the Dynamic Groups panel, click the **Browse Groups** button to display the **Browse Groups** dialog.
2. Select the dynamic group you want to copy by selecting a dynamic group from the list in the **Browse Dynamic Groups** dialog.
3. In the Dynamic Groups panel, edit the **Group name** text field with the desired name of the new group.
4. To save the new group click on the **Save Group** button to finish copying the group.

Deleting a Dynamic Group

To delete a dynamic group, do the following:

1. In the Dynamic Groups panel, click the **Browse Groups** button to display the **Browse Groups** dialog.
2. In the Browse Dynamic Groups dialog, select the dynamic group you want to delete.
3. In the Dynamic Groups panel, click on the **Delete Group** button to delete the chosen dynamic group.

Defining a SELECT Query String

You can write your own SQL queries to determine specific dynamic grouping criteria. By specifying the resource attribute in the SQL query, you can create a group on which a command can be run. These attributes are the node definitions in the IBM Cluster System Management database. Only persistent attributes can be used in a SELECT Query. To determine persistent attributes, type the following command at the command line:

```
lsrsrcdef -t -a p <resource class or a resource> | awk '{print $1}' | xargs -n3
```

The persistent attributes are listed in a three-column table. To list or test the **WHERE** clause of the SQL query string, type the following command at the command line:

```
lsnode -w "query_string"
```

To create a dynamic group, type:

```
nodegrp -w "query_string"
```

The *query_string* is what you specify in the SELECT query text area of the DCEM GUI in Dynamic Grouping panel.

Discovering Attributes Available for Queries:

- To list all persistent attributes for resource IBM.ManagedNode, type the following command at the command line.

  ```
  lsrsrcdef -t -a p IBM.ManagedNode | awk '{print $1}' | xargs -n3
  ```

  The following output displays:

```
Resource                                program_name                    Hostname
Macaddr                                 HWType                          HWModel
```

```
HWSerialNum                LParID                 ConsolePortNum
ConsoleServerName          HWControlPoint         SvcProcName
OSType                     OSVersion              OSDistribution
OSKernel                   InstallDiskType        InstallDisk
InstallMethod              UniversalId            ConsoleMethod
ConsoleServerNumber        PowerMethod
```

- To list all available attributes of managed nodes, type the following command at the command line. Only persistent attributes can be used in creating a SQL query to determine a dynamic group.

  ```
  lsnode -Al
  ```

  Output similar to the following displays:

  ```
  Hostname = endive.austin.ibm.com

  OSVersion =

  UniversalId = 158933068
  .
  .
  .
  ConfigChanged = 0
  Status = 1
  OST
  ```

- To list dynamic attributes that belong to the IBM.ManagedNode object, type the following command at the command line.

  ```
  lsrsrc -a d IBM.ManagedNode
  ```

  Output similar to the following displays:

  ```
  Resource Dynamic Attributes for: IBM.ManagedNode

  resource 1:

  PowerStatus = 127

  Status = 1

  ConfigChanged = 0
  ```

Examples of Dynamic Grouping SQL Queries Usage:

- To list the names of all the nodes types such as *host name c54* and exclude *c54n01* host names, type the following in the SELECT Query text area in the GUI:

  ```
  Hostname like 'c54%' && Hostname != 'c54n01.ppd.pok.ibm.com'
  ```

- To list the names of all the node types with a special power control, type the following in the SELECT Query text area in the GUI:

  ```
  PowerMethod == 'netfinity'
  ```

- To list all node names with the operating system type *linux*, type the following in the SELECT Query text area in the GUI:

  ```
  OSType like 'Linux%'
  ```

  The **OSType** attribute is defined during installation and is not a required attribute. If you did define the **OSType** attribute during the node installation, the above examples could be useful.

- To list all node names 'soft' or that have PowerMethod equal to 0, type:

  ```
  Hostname like 'soft%' || PowerMethod == '0'
  ```

For more detailed query and command syntax, see the Using Expressions section of this book and the *IBM Cluster Systems Management for Linux Technical Reference*.

Command Output and Activity Logs

DCEM command output and activity are saved in log files. Log files are stored in the following directory:

home/dcem/log/*log file name*

home is the home directory of the user under whose name the distributed command is run. *log file name* is the name of the log file containing the **dcem** command activity. All DCEM command activity of failures and successes are saved in this log file.

The default log file name is `dcem1.log`. The default maximum size for a log file is 10M. When `dcem1.log` is full, it is renamed to `dcem2.log`. New log entries are always written to `dcem1.log`.

The following is an example of DCEM log file contents:

```
TIME:     Sep 20 18:55:57.076
INFO:     Command Name:listing
Command: ls -l
Successful Machines: wsm14 wsm12 wsm06 wsm04 wsm15 wsm03
Failed Machines: wsm01 wsm00


TIME:     Oct 08 11:32:10.868
INFO:     Command Name:DiskSpace
command: df
Successful Machines: endive.austin.ibm.com
Failed Machines: westwing.austin.ibm.com
```

Diagnosing Problems with Distributed Command Execution Manager

DCEM uses IBM Cluster Systems Management (CSM), which, in turn, uses several other tools. Understanding this relationship can be helpful in diagnosing problems.

Problems Due to Insufficient Setup of Underlying Subsystems

The underlying CSM uses the Resource Monitoring and Control (RMC) subsystem to monitor and communicate with all nodes. If you are experiencing problems communicating with managed nodes, verify that RMC is running on each node, and that the security access and control list (ACL) file has been set up to allow the nodes to communicate with the management server.

DCEM uses the CSM **dsh** command to run commands on the nodes. In order for the **dsh** command to work, security needs to be set up on each node in such a way that **dsh** is allowed to run commands on that node. The security setup is dependent upon the type of remote shell you are using. The default remote shell is **rsh**, and to set up security on each node to allow dsh to run commands on that node (using **rsh**), you must add the management server host name to the /.rhosts file on the nodes that will be managed nodes. For example, if you want to run commands as root on machine2 from machine1, to the /.rhosts file on machine2, you would have to add the line `machine1 root`.

To verify the successful installation of CSM, list the active nodes by running the **lsnode -p** command and verify that **dsh** is working by running the **dsh -a date** command. For information, see the *IBM Cluster Systems Management for Linux Planning and Installation Guide*.

Interactive Commands and GUI applications

The CSM **dsh** command does not support the execution of interactive commands. Therefore, attempting to run an interactive command (one that requires input from standard in) from DCEM will not work.

To run an XWindows GUI application from DCEM, make sure that the DISPLAY variable is first set to your system's DISPLAY address, so that the GUI will display on your system. (For example, in the General panel, command area, you could first export the DISPLAY variable to your display's address prior to

issuing your command name.) If you run a GUI application correctly from DCEM, the application will remain in the ″Working″ state until you choose to exit the GUI.

Security Considerations and Remote Shells

DCEM takes in the same underlying security considerations as the CSM **dsh** command. You can use any underlying remote shell, but it is the system administrator's responsibility to configure and enable remote shell access. DCEM uses the CSM **dsh** command, which uses the underlying **rsh** security protocol by default. For more information about security considerations for **dsh** and preparing for **dsh** and configuring the remote shell, see the *IBM Cluster Systems Management for Linux Planning and Installation Guide*.

Diagnostic Information

All DCEM command activity of failures and successes are saved in log files to use later if you have to diagnose problems. These log files are stored in the following directory:

home/dcem/log/*log file name*

To see more detail on the actual underlying CSM command execution string specified as a result of running your created command specification, run the Perl script (outside of DCEM) in debug mode and directly from your AIX command line, as follows:

```
<commandSpecificationName>.pl -debug
```

Diagnostic Information in Related Publications

Diagnosing DCEM problems relies heavily on being able to diagnose problems in the underlying CSM and RMC subsystems. For more information on diagnosing problems in DCEM's underlying subsystems, refer to the Diagnostic Information section of this book.

Example of a saved command script

```
#!/usr/bin/perl -w
##############################################################################
#
# Licensed Materials - Property of IBM
#
# (C) COPYRIGHT International Business Machines Corp. 1994,2001
# All Rights Reserved
#
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
##############################################################################

##############################################################################
#
# Example perl file -
# Run via:
#   perl <this-perl-script.pl>  [-debug] [-non_interactive]
# E.g.
#   perl listusers.pl
#   perl listusers.pl -debug
#   perl listusers.pl -non_interactive
#
# Author : Generated by Distributed Command Execution Manager
#
##############################################################################

##############################################################################
# perl information
#
##############################################################################

$| = 1;           # Flush output buffer
require 5.003;    # need this version of Perl or newer
```

```
use English;      # use English names, not cryptic ones
use FileHandle;   # use FileHandles instead of open(),close()
use Carp;         # get standard error / warning messages
use strict;       # force disciplined use of variables

#############################################################################
# GLOBAL VARIABLES AND CONSTANTS
#
#############################################################################

my ($TRUE)                                = "TRUE";
my ($FALSE)                               = "FALSE";

# ---------------------------------------------------------------------------
# Command Environment Variables
# ---------------------------------------------------------------------------

my (@HOSTS)                               = ('b905em17.austin.ibm.com');
my (@GROUPS)                              = ('Group1','Group2');

# ---------------------------------------------------------------------------
# Command Specification
# ---------------------------------------------------------------------------

my ($CMD_SPECIFICATION)=<<'END_CMD_SPECIFICATION'
ls -l; whoami; pwd; ls -l
END_CMD_SPECIFICATION
;

# ---------------------------------------------------------------------------
# Script options and user default settings
# NOTE: You must add any new options to
# the OPTION_FLAGS array.
# ---------------------------------------------------------------------------

my ($DEBUG_FLAG)               = "-debug";
my ($LAUNCH_GUI_FLAG)          = "-gui";
my ($FORMAT_OUTPUT_FLAG)       = "-format_output";
my ($PROMPT_USER_FLAG)         = "-non_interactive";
my (@OPTION_FLAGS)             = ($DEBUG_FLAG,
                                  $LAUNCH_GUI_FLAG,
                                  $FORMAT_OUTPUT_FLAG,
                                  $PROMPT_USER_FLAG);

my ($DEBUG)                    = $FALSE;
my ($LAUNCH_GUI)               = $FALSE;
my ($FORMAT_OUTPUT)            = $FALSE;
my ($PROMPT_USER)              = $TRUE;

# ---------------------------------------------------------------------------
# Csm Distributed Services
# ---------------------------------------------------------------------------

my ($DISTRIB_SERVICE)                     = "/opt/csm/bin/dsh";
my ($DISTRIB_POST_PROCESSING_COMMAND)     = "/opt/csm/bin/dshbak";
my ($DISTRIB_DEFAULT_REMOTE_SHELL)        = "rsh";

# ---------------------------------------------------------------------------
#      Dsh Options
# ---------------------------------------------------------------------------

my ($DISTRIB_HOST_OPTION)                 = "-n";
my ($DISTRIB_GROUP_OPTION)                = "-N";
my ($DISTRIB_FANOUT_OPTION)               = "-f";
my ($DISTRIB_STREAMING_OPTION)            = "-s";
my ($DISTRIB_VERIFY_HOSTS_OPTION)         = "-v";
my ($DISTRIB_USER_OPTION)                 = "-l";
```

```
my ($DISTRIB_REMOTE_SHELL_OPTIONS_OPTION)  = "-o";
my ($DISTRIB_REMOTE_SHELL_PATH_OPTION)     = "-r";

# ----------------------------------------------------------------------
# Additional Csm Command Environment Variables
# ----------------------------------------------------------------------

my ($FANOUT)                               = 64;
my ($STREAMING)                            = $TRUE;
my ($VERIFY_HOSTS)                         = $FALSE;
my ($USER)                                 = "root";
my ($REMOTE_SHELL_OPTIONS)                 = "";
my ($REMOTE_SHELL)                         = "rsh";
my ($COMMAND_PATH)                         = '$PATH';

# ----------------------------------------------------------------------

##########################################################################
# Sub Functions
##########################################################################

##########################################################################
# This sub-function displays the string passed to it.
# This sub-function should be used only for debug messages.
#
# @param the messages string to be displayed
#
##########################################################################

sub debug_message ($)
{
    if ($DEBUG eq $TRUE)
    {
        display_message (@_);
    }
}

##########################################################################
# This sub-function displays the string passed to it.
# This sub-function should be used to convey information to
# users
#
# @param the messages string to be displayed
#
##########################################################################

sub display_message ($)
{
    my ($str) = @_;
    print "$str";
}

##########################################################################
# This sub-function executes the command.
#
##########################################################################

sub run_distributed_command_line ()
{
    debug_message ("Enter sub-function run_distributed_command()...\n");

    my (@execution_string);
    my ($line) = "";

    # Get the arguments

    # construct the execution string based on the parameters
```

```
    @execution_string = build_execution_string();

    # run the command
    debug_message ("Running the command: @execution_string \n\n");

    my (@output);
    my ($current_pid) = fork;
    if ($current_pid == 0) {
        @output = exec (@execution_string);
    }
    elsif ($current_pid) {
        debug_message("In parent process, before wait.\n");
        my ($child_pid) = wait;
        debug_message("In parent process, after wait. Child pid was $child_pid.\n");
    }
    else {
        die "fork error: $!\n";
    }

    foreach $line (@output)
    {
        display_message ($line);
    }

    debug_message ("\nLeave sub-function run_distributed_command().\n");
    return (@output);
}

#############################################################################
# This sub-function invokes the
# Distributed Command Execution Manager GUI
#
#############################################################################

sub run_distributed_command_gui ()
{
    debug_message ("Enter sub-function run_distributed_command_gui()...\n");

    my ($cmd_name) = $0;
    $cmd_name =~ s/\.pl$//;
    debug_message( "Command name to load is $cmd_name\n");

    '/opt/csm/dcem/bin/dcem -command $cmd_name';

    debug_message ("Leave sub-function run_distributed_command_gui().\n");
}

#############################################################################
# This sub-function gets the path to the selected remote shell.
#
#-- This is generated by the printBuildCommandLineFunction() method
#############################################################################

sub get_remote_shell_path ()
{
    debug_message ("Enter sub-function get_remote_shell_path()...");

    my ($execution_string) = "ksh -c \"which $REMOTE_SHELL\"";
    my ($line);

    # run the command
    debug_message ("Running the which $REMOTE_SHELL command: $execution_string \n\n");

    my (@output) = '$execution_string';

    $line = $output[0];
    my (@splitLine) = split(' ', $line);
```

```
    my ($path) = $splitLine[0];

    if ($line =~ /$REMOTE_SHELL$/)
    {
        debug_message ("Found remote shell path $path\n");
        return $path;
    }
    else
    {
        display_message ("The remote shell $REMOTE_SHELL was not found. Dsh will use the
        default $DISTRIB_DEFAULT_REMOTE_SHELL remote shell when executing this command.\n");
        return "";
    }
}

#########################################################################
# This sub-function contructs the complete execution string.
#
#########################################################################

sub build_execution_string ()
{
    debug_message ("Enter sub-function build_execution_string()...\n");

    my ($i) = 0;
    my ($cmd_path) = "";
    my (@execution_string);

    $execution_string[$i] = $DISTRIB_SERVICE;
    $execution_string[++$i] = $DISTRIB_FANOUT_OPTION;
    $execution_string[++$i] = $FANOUT;
    debug_message ("Execution string is: @execution_string\n");

    if (($STREAMING eq $TRUE) && ($FORMAT_OUTPUT eq $FALSE))
    {
        $execution_string[++$i] = $DISTRIB_STREAMING_OPTION;
        debug_message ("Execution string is: @execution_string\n");
    }

    $execution_string[++$i] = $DISTRIB_USER_OPTION;
    $execution_string[++$i] = $USER;
    debug_message ("Execution string is: @execution_string\n");

    if ($REMOTE_SHELL ne $DISTRIB_DEFAULT_REMOTE_SHELL)
    {
        my ($remote_shell_path) = get_remote_shell_path();
        if ($remote_shell_path ne "")
        {
            $execution_string[++$i] = $DISTRIB_REMOTE_SHELL_PATH_OPTION;
            $execution_string[++$i] = $remote_shell_path;
            debug_message ("Execution string is: @execution_string\n");

            $execution_string[++$i] = $DISTRIB_REMOTE_SHELL_OPTIONS_OPTION;
            $execution_string[++$i] = $REMOTE_SHELL_OPTIONS;
            debug_message ("Execution string is: @execution_string\n");
        }
    }
    elsif ($REMOTE_SHELL_OPTIONS ne "")
    {
        $execution_string[++$i] = $DISTRIB_REMOTE_SHELL_OPTIONS_OPTION;
        $execution_string[++$i] = $REMOTE_SHELL_OPTIONS;
        debug_message ("Execution string is: @execution_string\n");
    }

    if (@GROUPS)
    {
        $execution_string[++$i] = $DISTRIB_GROUP_OPTION;
```

```
        $execution_string[++$i] = join(",", @GROUPS);
        debug_message ("Execution string is: @execution_string\n");
    }

    if ($VERIFY_HOSTS eq $TRUE)
    {
        $execution_string[++$i] = $DISTRIB_VERIFY_HOSTS_OPTION;
        debug_message ("Execution string is: @execution_string\n");
    }

    if (@HOSTS)
    {
        $execution_string[++$i] = $DISTRIB_HOST_OPTION;
        $execution_string[++$i] = join(",", @HOSTS);
        debug_message ("Execution string is: @execution_string\n");
    }

    if ($COMMAND_PATH ne '')
    {
        $cmd_path = "export PATH=$COMMAND_PATH;";
        debug_message ("Command path string is: $cmd_path\n");
    }

    $execution_string[++$i] = join(" ", $cmd_path, $CMD_SPECIFICATION);
    debug_message ("Execution string is: @execution_string\n");

    if ($FORMAT_OUTPUT eq $TRUE)
    {
        $execution_string[++$i] = join("", " | ", $DISTRIB_POST_PROCESSING_COMMAND);
        debug_message ("Execution string is: @execution_string\n");
    }

    debug_message ("Leave sub-function build_execution_string().\n");
    return (@execution_string);
}

######################################################################
# This sub-function asks the user whether the program should
# continue or not.
#
# @param cmd_spec - the command specification
# @param hosts    - the host machines to run the command on
# @param groups   - the groups to run the command on
#
######################################################################

sub confirm_command_execution ($$$)
{
    debug_message ("Enter sub-function confirm_command_execution()...\n");

    my ($cmd_spec, $hosts_ref, $groups_ref) = @_;
    my (@hosts) = @$hosts_ref;
    my (@groups) = @$groups_ref;
    my ($host);
    my ($group);
    my ($reply) = "";

    display_message("The command \"$cmd_spec\" ");

    if ((scalar(@hosts) == 0) && (scalar(@groups) == 0))
    {
        display_message("has no targets specified.\n");
        return($FALSE);
    }

    display_message ("is about to be executed on the following ");
```

```
    if (scalar(@hosts))
    {
        display_message( "hosts:\n\t");

        foreach $host (@hosts)
        {
            display_message ("$host ");
        }
        display_message ("\n");

        if (scalar(@groups))
        {
            display_message ("and ");
        }
    }

    if (scalar(@groups))
    {
        display_message ("groups:\n\t");

        foreach $group (@groups)
        {
            display_message ("$group ");
        }
        display_message ("\n");
    }

    while (defined($reply) && $reply !~ /[yYnN]/ )
    {
        display_message ("Do you wish to continue (y/n)?: ");
        $reply = <STDIN>;
        chop ($reply);
    }

    debug_message ("Leave sub-function confirm_command_execution().\n");

    # Check the reply to determine whether to continue
    if ($reply =~ /[yY]/)
    {
        return($TRUE);
    }
    else
    {
        return ($FALSE);
    }
}

#############################################################################
# This sub-function exits the program with an appropriate
# exit code.
#
# @param
#
#############################################################################

sub exit_program ($$)
{
    debug_message ("Enter sub-function exit_program()...\n");

    my ($msg, $exit_code) = @_;

    display_message ($msg);

    debug_message ("Leave sub-function exit_program().\n");
    debug_message ("Exiting program with exit_code: $exit_code\n");
    exit ($exit_code);
}
```

```
#########################################################################
# This sub-function display the usage message for this command
#
# @param
#
#########################################################################

sub usage ($)
{
    debug_message ("Enter sub-function usage()...\n");

    my ($bad_option) = @_;
    display_message ($bad_option);

    display_message ("Usage: perl $0 [$DEBUG_FLAG] [$PROMPT_USER_FLAG]\n\n");

    display_message (" $DEBUG_FLAG\t\t\t- displays debug messages\n");
    display_message (" $PROMPT_USER_FLAG\t- does not prompt user for input\n");

    exit_program("", 1);

    debug_message ("Leave sub-function usage().\n");
}

#########################################################################
# This function logs entries into a file whose name is
# provided as the imput argument.
#
# @param - log file name
#
#########################################################################

sub generate_log_entries
{
    debug_message ("Enter sub-function generate_log_entries()...\n");
    '/opt/csm/dcem/bin/dLogMgr -s \"@_\"';

    debug_message ("Leave sub-function generate_log_entries().\n");
}

#########################################################################
# This function mails status reports after the command is
# executed
#
# @param - email addresses
#
#########################################################################

sub mail_report ($)
{
    debug_message ("Enter sub-function mail_report()...\n");

    debug_message ("Leave sub-function mail_report().\n");
}

#########################################################################
# This sub-function check all the options, in the
# @ARGV array.
# Flags are assumed to begin with a '-' (dash or minus sign).
#
#########################################################################

sub check_options ()
{
    debug_message ("Enter sub-function check _options ()...\n");
```

```
    my ($CMD_OPTION) = "";
    my ($TMP_CMD_OPTION) = "";
    my ($OPTION_FLAG) = "";

    foreach $CMD_OPTION (@ARGV)
    {
        $TMP_CMD_OPTION = "";

        # Check for incomplete and/or ambigous options
        foreach $OPTION_FLAG (@OPTION_FLAGS)
        {
            if (index ($OPTION_FLAG, $CMD_OPTION) ==  0)
            {
                if ($TMP_CMD_OPTION eq "")
                {
                    $TMP_CMD_OPTION = $OPTION_FLAG;
                }
                else
                {
                    $TMP_CMD_OPTION = "AMBIGUOUS"; #ambiguous
                    last;
                }
            }
        }

        if ($TMP_CMD_OPTION eq $PROMPT_USER_FLAG)
        {
            debug_message ("Setting PROMPT_USER to FALSE.\n");
            $PROMPT_USER = $FALSE;
        }
        elsif ($TMP_CMD_OPTION eq $LAUNCH_GUI_FLAG)
        {
            debug_message ("Setting LAUNCH_GUI to TRUE.\n");
            $LAUNCH_GUI = $TRUE;
        }
        elsif ($TMP_CMD_OPTION eq $FORMAT_OUTPUT_FLAG)
        {
            debug_message ("Setting FORMAT_OUTPUT to TRUE.\n");
            $FORMAT_OUTPUT = $TRUE;
        }
        elsif ($TMP_CMD_OPTION eq $DEBUG_FLAG)
        {
            $DEBUG = $TRUE;
            debug_message ("Setting DEBUG to TRUE.\n");
        }
        elsif ($TMP_CMD_OPTION eq "AMBIGUOUS")
        {
            usage ("Ambiguous option: $CMD_OPTION\n\n");
        }
        else
        {
            usage ("Error! Bad option: $CMD_OPTION\n\n");
        }
    }

    debug_message ("Leave sub-function check _options ().\n");
}

#########################################################################
# This sub-function invokes all other subfunctions and is
# responsible for executing the command.
#
#########################################################################

sub main_driver ()
{
    # Check if any command line arguments have been passed in
```

```
    check_options ();

    debug_message ("Enter function main_driver()...\n");

    my ($host) ="";
    my ($group) ="";
    my ($continue_program) = "TRUE";
    my ($error_code) = 0;

    debug_message ("hosts:\n");
    foreach $host (@HOSTS)
    {
        debug_message ("\t$host\n");
    }

    debug_message ("groups:\n");
    foreach $group (@GROUPS)
    {
        debug_message ("\t$group\n");
    }

    # What about the GUI option........
    if ($LAUNCH_GUI eq $TRUE)
    {
        display_message ("Launching GUI...\n");

        run_distributed_command_gui();

        exit_program ("", 0 );
    }

    debug_message ("User prompt setting is: $PROMPT_USER\n");

    if ($PROMPT_USER eq $TRUE)
    {
        $continue_program = confirm_command_execution ($CMD_SPECIFICATION,
                            \@HOSTS, \@GROUPS);
        if ($continue_program eq $FALSE)
        {
            my ($exit_msg) = "Program exited without executing command.\n";
            exit_program ($exit_msg, 1 );
        }
    }

    # After all the GUI and PROMPT options have been processed,
    # we are now ready to run the script
    my (@results) = run_distributed_command_line ();
    if (-e "/opt/csm/dcem/bin/dLogMgr")
    {
        generate_log_entries(@results);
    }

    debug_message ("Leave function main_driver().\n");
}

#########################################################################
# This the start of the script.
#
#########################################################################

main_driver ();

0;  # return 0 (no error from this script)

#########################################################################
```

```
#----------------- This is the END of the Script --------------
#
#####################################################################
```

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 Road
Poughkeepsie, New York 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 terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

- IBM, AIX, AIX 5L, Netfinity, RS/6000, SP, and the e (logo) are trademarks or registered trademarks of International Business Machines Corporation.
- Equinox is a trademark of Equinox Systems, Inc.
- Linux is a registered trademark of Linus Torvalds.
- Myrinet is a trademark of Myricom, Inc.
- Red Hat and RPM are trademarks of Red Hat, Incorporated.
- Java and all Java-based trademarks and logos are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Other company, product, or service names may be the trademarks or service marks of others.

Publicly Available Software

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

Advanced TFTP	Advanced TFTP (atftp) is a client/server implementation of the TFTP protocol. It is licensed under the GNU General Public License (GNU GPL), which can be found at **http://www.gnu.org/copyleft/gpl.html**.
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 or ″reverse-telnet″ to console servers or both.
DBD-CSV, Text-CSV_XS	Licensed by GPL or Artistic, these are dynamically loaded Perl modules.
DBI	Licensed by GPL or Artistic, this is a dynamically loaded Perl module.
fping	Copyrighted by Stanford University, this is executed as a separate binary.
Kerberos	Provides authentication of the execution of remote commands.
Perl	Practical Extraction and Report Language is licensed under the Artistic license.
Perl-to-C extensions	Practical Extraction and Report Language-to-C extensions is distributed 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.
SYSLinux	SYSLinux includes PXELINUX, which CSM uses to control the behavior of network boots. SYSLinux is licensed under the GNU GPL.

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

Note: The distribution for these products includes the source code and associated documentation. All copyright notices and license conditions in the documentation must be respected. You can find version and distribution information for each of these products in the *Specified Operating Environment* section of the *IBM Cluster Systems Management for Linux Planning and Installation Guide*. For these non-IBM products, the following license terms apply in lieu of the International Program License Agreement.

The inclusion herein of copies of various licenses is not meant to imply endorsement of the principles, methodologies, or views that are contained therein, either express or implied.

For the fping open source code, see the following URL: **http://www.fping.com** (The license is imbedded in the source code.)

For the fping open source code, the following terms apply:

```
Copyright (c) 1991, 1994, 1997 Board of Trustees
              Leland Stanford Jr. University
```

Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by Stanford University. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ″AS IS″ AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

For the DBD-CSF, DBI, Text-CSV_XS and SQL-Statement open source code licensed under the ″Artistic License,″ the following terms apply:

```
Definitions

"Package" refers to the collection of files distributed by the
```

```
Copyright Holder, and derivatives of that collection of files created
through textual modification.

"Standard Version" refers to such a Package if it has not been
modified, or has been modified in accordance with the wishes of the
Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights
for the package.

"You" is you, if you're thinking about copying or distributing this
Package.

"Reasonable copying fee" is whatever you can justify on the basis of
media cost, duplication charges, time of people involved, and so on.
(You will not be required to justify it to the Copyright Holder, but
only to the computing community at large as a market that must bear the
fee.)

"Freely Available" means that no fee is charged for the item itself,
though there may be fees involved in handling the
item. It also means that recipients of the item may redistribute
it under the same conditions they received it.
```

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that all copies contain the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
   a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
   b. use the modified Package only within your corporation or organization.
   c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.
   d. make other distribution arrangements with the Copyright Holder.
4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:
   a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.
   b. accompany the distribution with the machine-readable source of the Package with your modifications.
   c. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
   d. make other distribution arrangements with the Copyright Holder.
5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of

yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whomever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called ″undump″ or ″unexec″ methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.
7. C subroutines (or comparably compiled subroutines in other languages supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.
8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.
9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.
10. THIS PACKAGE IS PROVIDED ″AS IS″ AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

For the Advanced TFTP (atftp), cfengine, and SYSLinux code, licensed under the GNU GENERAL PUBLIC LICENSE Version 2, June 1991, see the following.

```
Copyright (c) 1989, 1991 Free Software Foundation, Inc. 59 Temple
Place, Suite 330, Boston, MA 02111-1307 USA
```

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The ″Program″, below, refers to any such program or work, and a ″work based on the Program″ means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term ″modification″.) Each licensee is addressed as ″you″.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program. You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.
2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work

based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

      The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

      If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.
4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances. It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and ″any later version″, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Index

P

R

S

T

U

V

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

IBM Cluster Systems Management for Linux®
Administration Guide
Version 1.1

Publication No. SA22-7873-01

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
SA22-7873-01

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

Cut or Fold
Along Line

Program Number: 5765–E88

SA22-7873-01