

IBM Reliable Scalable Cluster Technology



Administration Guide

IBM Reliable Scalable Cluster Technology



Administration Guide

Note

Before using this information and the product it supports, read the information in "Notices" on page 197.

Seventh Edition (November 2004)

This edition applies to:

- version 5, release 2 of IBM AIX 5L for POWER™ (product number 5765-E62) with the 5200-04 Recommended Maintenance package
- version 5, release 3 of IBM AIX 5L for POWER (product number 5765-G03)
- version 1, release 4 of IBM Cluster Systems Management (CSM) for Linux on POWER (product number 5765-G16)
- version 1, release 4 of IBM Cluster Systems Management (CSM) for Linux on xSeries and @server325 (product number 5765-E88)
- version 1, release 2 of IBM Tivoli System Automation for Multiplatforms (product numbers 5639-N53 and 5655-I53)

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

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

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

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

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

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

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

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

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

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

© Copyright International Business Machines Corporation 2002, 2004. All rights reserved.

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

Contents

	About this book	vii
	Who should use this book	vii
	How this book is organized	vii
	Conventions and terminology used in this book	viii
	Prerequisite and related information	ix
I	Using LookAt to find message explanations	x
	How to send your comments	x
	Chapter 1. What is RSCT?	1
	What are management domains and peer domains?	1
	What is RMC?	2
	What are the RSCT resource managers?	3
	What are the cluster security services?	4
	What are Topology Services?	4
	What are Group Services?	5
	What are IBM Virtual shared disks and IBM Recoverable virtual shared disks?	5
	What is the low-level application programming interface (LAPI)?	6
	What is the System Resource Controller (SRC)?	6
	Chapter 2. RSCT installation and software verification	9
I	RSCT installation verification on AIX nodes	9
I	RSCT installation verification on Linux nodes	11
I	Required RSCT fix for Red Hat AS 2.1 on x86	14
I	Applying required patch for Red Hat 8.0 on x86	14
I	Applying required patch for Red Hat 9.0 on x86	15
I	Kernel requirement for Red Hat EL 3.0 on AMD-64	15
I	Kernel requirement for SUSE SLES 8 Linux on pSeries	15
I	Kernel requirement for SUSE SLES 8 Linux on iSeries	15
I	Obtaining compat-libstdc++ for Red Hat EL 3 on Power	15
I	Kernel update requirement for SUSE SLES 9 Linux on all platform	15
I	Supported Linux distributions for RSCT 2.4.1.0	16
	Chapter 3. Creating and administering an RSCT peer domain	19
	What is an RSCT peer domain?	19
	What is the configuration resource manager?	19
	What are communication groups?	19
	What is quorum?	20
	What can I do using configuration resource manager commands?	22
	Prerequisites and restrictions to using configuration resource manager commands	23
	Supported RSCT versions	24
I	RSCT Support for VIPA	24
	Migration	24
	Creating a peer domain	25
	Step 1: prepare initial security environment on each node that will participate in the peer domain	26
	Step 2: create a new peer domain	27
	Step 3: bring the peer domain online	29
	Adding nodes to an existing peer domain	32
	Step 1: prepare security environment on the node	32
	Step 2: add node to the peer domain	34
	Step 3: bring node online in the peer domain	35
	Taking individual nodes of a peer domain, or an entire peer domain, offline	36

Taking a peer domain node offline	36
Taking a peer domain offline	37
Removing individual nodes from, or removing an entire, peer domain	37
Removing a node from a peer domain	37
Removing a peer domain	38
Changing a peer domain's quorum type	39
Understanding and working with communication groups	39
Listing communication groups	40
Modifying a communication group's characteristics	41
Manually configuring communication groups	43
Modifying Topology Services and Group Services parameters	46
Changing IP addresses in a peer domain	47
Determining how your system responds to domain partitioning and subsystem daemon failure	48
Setting the critical resource protection method for a peer domain or a node in a peer domain	49
Overriding the configuration resource manager's operational quorum calculation to force operational quorum	51
Determining how the configuration resource manager will resolve tie situations when calculating operational quorum	52
Chapter 4. Managing and monitoring resources using RMC and resource managers	61
Understanding RMC and resource managers	62
What is RMC?	62
What is a resource manager?	63
How does RMC and the resource managers enable you to manage resources?	70
How do RMC and the resource managers enable you to monitor resources?	70
How does RMC implement authorization?	75
How do I determine the target nodes for a command?	75
Managing user access to resources using RMC ACL files	77
Format of an ACL file	77
Listing Resource Information	79
Listing available resource classes	80
Displaying attribute value information for a resource or a resource class	81
Displaying attribute definition information for a resource or a resource class	83
Basic resource monitoring	84
Listing conditions, responses, and condition/response associations	85
Creating a condition/response association	88
Starting condition monitoring	90
Stopping condition monitoring	91
Removing a condition/response association	92
Using the audit log to track monitoring activity	93
Advanced resource monitoring	98
Creating, modifying and removing conditions	98
Creating, modifying, and removing responses	112
Querying CIM properties	122
Catching SNMP traps on Linux nodes	127
Locking and unlocking conditions, responses, and condition/response links	128
Using expressions to specify condition events and command selection strings	130
SQL restrictions	132
Supported base data types	132
Structured data types	133
Data types that can be used for literal values	133
How variable names are handled	135

|
|
|
|

Operators that can be used in expressions	135
Pattern matching.	139
Examples of expressions.	139
Chapter 5. Controlling access to root commands and scripts	141
Overview of LP resource manager operation	141
Determining the target nodes for an LPRM command	142
Monitoring LP resources and operations	143
Steps for defining LP resources and authorized users	143
Step for running an LP resource	144
Steps for changing an LP resource	145
Steps for removing LP resources.	145
Chapter 6. Understanding and administering cluster security services	147
Understanding cluster security services' authentication.	147
Understanding credentials based authentication	148
Understanding Host Based Authentication	148
Understanding cluster security services' authorization	150
Understanding native identity mapping.	150
Cluster security services administration	151
Configuring the cluster security services library	151
Configuring the Host Based Authentication mechanism.	152
Configuring the global and local authorization identity mappings	161
Chapter 7. The Topology Services subsystem	167
Introducing Topology Services	167
Topology Services components	168
The Topology Services daemon (hatsd)	168
Pluggable NIMs	170
Port numbers and sockets	170
The cthatsctrl control command	171
The cthats startup command	171
The cthatstune tuning command	171
Files and directories	172
Components on which Topology Services depends	173
Configuring and operating Topology Services	174
Setting Topology Services tunables	174
Configuring Topology Services.	175
Initializing Topology Services daemon	176
Operating Topology Services daemon	177
Topology Services procedures.	182
Displaying the status of the Topology Services daemon	182
Chapter 8. The Group Services subsystem	185
Introducing Group Services	185
Group Services components	186
The Group Services daemon (hagsd)	186
The Group Services API (GSAPI)	187
Port numbers and sockets	187
The cthagctrl control command	188
Files and directories	188
Components on which Group Services depends	189
Configuring and operating Group Services	189
Configuring Group Services.	189
Initializing Group Services daemon	190
Group Services initialization errors	192

Group Services daemon operation	192
Group Services procedures	192
Displaying the status of the Group Services daemon	192
Appendix. Product-related information.	195
RSCT version	195
Accessibility	195
Using assistive technologies	195
ISO 9000	195
Product-related feedback.	195
Notices	197
Trademarks.	199
Glossary	201
Index	205

About this book

This book describes various component subsystems of IBM's Reliable Scalable Cluster Technology (RSCT). On AIX, the RSCT components are included as part of the AIX 5L operating system. The RSCT components are also available as part of various Linux-based products such as IBM Cluster Systems Management (CSM) for Linux and IBM Tivoli® System Automation for Multiplatforms. This book describes:

- the Resource Monitoring and Control (RMC) subsystem and core resource managers that together enable you to monitor various resources of your system and create automated responses to changing conditions of those resources.
- how to use the configuration resource manager to configure a set of nodes into a cluster for high availability. Such a cluster is called an *RSCT peer domain*.
- the basics of cluster security services which are used by other RSCT components and other cluster products for authentication. This book describes some common administration tasks associated with the cluster security services.
- the Topology Services subsystem which provides other subsystems with network adapter status, node connectivity information, and a reliable messaging service.
- the Group Services subsystem which provides other component subsystems with a distributed coordination and synchronization service.

Reliable Scalable Cluster Technology (RSCT) is a component of the following:

- AIX 5L
- Cluster Systems Management (CSM) for Linux
- IBM Tivoli System Automation for Multiplatforms

Who should use this book

This book should be read by anyone who wants to:

- understand the core RSCT components.
- configure a set of nodes into an RSCT peer domain.
- Understand how authentication is handled by cluster security services, and administer cluster security.
- Understand, and diagnose problems with, Topology Services.
- Understand, and diagnose problems with, Group Services.

How this book is organized

This book is divided into the following chapters:

- Chapter 1, "What is RSCT?," on page 1 provides a high-level description of the various component subsystems of RSCT.
- Chapter 2, "RSCT installation and software verification," on page 9 describes how you can determine if the RSCT components are installed.
- Chapter 3, "Creating and administering an RSCT peer domain," on page 19 describes how to use configuration resource manager commands to create and administer an RSCT peer domain. It describes how to:
 - create a new peer domain
 - add nodes to an existing peer domain
 - create and modify a communication group. A communication group controls how liveness checks are performed between the communications resources within the peer domains

- take nodes of a peer domain, or an entire peer domain, offline
- remove individual nodes from, or remove an entire, peer domain
- Chapter 4, “Managing and monitoring resources using RMC and resource managers,” on page 61 describes how you can use RMC and core resource managers to detect conditions of interest in your machine and associated resources and automatically take action when those conditions occur. This chapter provides:
 - an overview of monitoring concepts
 - instructions on using Event Response Resource Manager (ERRM) commands to associate automatic responses with monitored conditions.
- Chapter 5, “Controlling access to root commands and scripts,” on page 141 describes how you can use the least privilege resource manager to enhance the security of commands and scripts that require root authority to run.
- Chapter 6, “Understanding and administering cluster security services,” on page 147 provides an overview of the security infrastructure that enables RSCT components to authenticate the identity of other parties. It provides information on administrative tasks associated with cluster security services.
- Chapter 7, “The Topology Services subsystem,” on page 167 provides an overview of, and describes how you can troubleshoot problems related to, the Topology Services subsystem.
- Chapter 8, “The Group Services subsystem,” on page 185 provides an overview of, and describes how you can troubleshoot problems related to, the Group Services subsystem.
- “Product-related information,” on page 195 contains some additional information on the products that contain the RSCT technology, how to determine the RSCT version, and how to provide feedback on RSCT.

This book should be read by anyone who wants to:

- understand the core RSCT components.
- configure a set of nodes into an RSCT peer domain.
- Understand how authentication is handled by cluster security services, and administer cluster security.
- Understand, and diagnose problems with, Topology Services.
- Understand, and diagnose problems with, Group Services.

Conventions and terminology used in this book

This book uses the following typographic conventions:

Convention	Usage
bold	Bold words or characters represent system elements that you must use literally, such as: command names, file names, flag names, and path names.
constant width	Examples and information that the system displays appear in constant-width typeface.
<i>italic</i>	<i>Italicized</i> words or characters represent variable values that you must supply. <i>Italics</i> are also used for book titles, for the first use of a glossary term, and for general emphasis in text.
{ <i>item</i> }	Braces indicate required items.

Convention	Usage
[<i>item</i>]	Brackets indicate optional items.
	<ol style="list-style-type: none"> 1. In the left margin of the book, vertical lines indicate technical changes to the information. 2. In syntax (or synopsis) statements, vertical lines are used as <i>pipe</i> characters. See <i>RSCT for AIX 5L: Technical Reference</i> or <i>RSCT for Linux: Technical Reference</i> for more information.
\	<p>In command examples, a backslash indicates that the command continues on the next line. For example:</p> <pre>mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \ -E "PercentTotUsed < 85" -m d "FileSystem space used"</pre>

See the “Glossary” on page 201 for definitions of some of the terms that are used in this book.

Prerequisite and related information

The core Reliable Scalable Cluster Technology (RSCT) publications are:

- *RSCT Administration Guide*, SA22-7889, provides an overview of the RSCT components and describes how to:
 - Create and administer RSCT peer domains.
 - Manage and monitor resources using the resource monitoring and control (RMC) subsystem.
 - Administer cluster security services for RSCT peer domains and CSM management domains.
- *RSCT Diagnosis Guide*, SA23-2202, describes how to diagnose and resolve problems related to the various components of RSCT. This book is a companion volume to *RSCT Messages*, which lists the error messages that may be generated by each RSCT component. While *RSCT Messages* describes the appropriate user responses to messages that are generated by RSCT components, this book contains additional and more detailed diagnostic procedures.
- *RSCT Messages*, GA22-7891, lists the error messages that may be generated by each RSCT component. For each message, this manual provides an explanation of the message, and describes how you should respond to it.
- *RSCT for AIX 5L: Technical Reference*, SA22-7890, and *RSCT for Linux: Technical Reference*, SA22-7893, provide detailed reference information about all of the RSCT commands, daemons, files, and scripts.

In addition to these core RSCT publications, the library contains the following publications of interest:

- *RSCT Group Services Programming Guide and Reference*, SA22-7888, contains information for programmers who want to write new clients that use the group services subsystem’s application programming interface (GSAPI) or who want to add the use of group services to existing programs. This book is intended for programmers of system management applications who want to use group services to make their applications highly available.
- *RSCT for AIX 5L: LAPI Programming Guide*, SA22-7936, provides conceptual, procedural, and reference information about the low-level application programming interface (LAPI). LAPI is part of the AIX® implementation of RSCT only; it is not available with RSCT for Linux™. LAPI is a message-passing API

that provides optimal communication performance on an IBM® @server pSeries® High Performance Switch (pSeries HPS).

- *RSCT for AIX 5L: Managing Shared Disks, SA22-7937*, describes the shared disk management facilities of IBM @server Cluster 1600 server processors — the optional virtual shared disk and recoverable virtual shared disk components of RSCT for AIX 5L™. These components are part of the AIX implementation of RSCT only; they are not available with RSCT for Linux. This book describes how you can use these components to manage cluster disks to enable multiple nodes to share the information they hold. The book includes an overview of the components and explains how to plan for them, install them, and use them to add reliability and availability to your data storage.

An **RSCT Documentation Updates** file is maintained on the World Wide Web at the following URL:

<http://publib.boulder.ibm.com/clresctr/docs/rsct/docupdates.html>

This file contains updates to the RSCT documentation. These updates include documentation corrections and clarifications, as well as information (such as needed software patches) that was discovered after the RSCT books were published. Please check the **RSCT Documentation Updates** file for pertinent information.

To access all RSCT documentation, refer to the **IBM @server Cluster Information Center**. This Web site, which is located at **<http://publib.boulder.ibm.com/clresctr>**, contains the most recent RSCT documentation in PDF and HTML formats.

The current RSCT books and earlier versions of the library are also available in PDF format from the **IBM Publications Center** Web site, which is located at **<http://www.ibm.com/shop/publications/order>**. It is easiest to locate a manual in the **IBM Publications Center** by supplying the manual's publication number. The publication number for each of the RSCT books is listed after the book title in the preceding list.

Using LookAt to find message explanations

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

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

www.ibm.com/eserver/zseries/zos/bkserv/lookat

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

How to send your comments

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

- Send your comments by e-mail to: mhvrdfs@us.ibm.com

Include the book title and order number, and, if applicable, the specific location of the information you have comments on (for example, a page number or a table number).

- Fill out one of the forms at the back of this book and return it by mail, by fax, or by giving it to an IBM representative.

To contact the IBM cluster development organization, send your comments by e-mail to: cluster@us.ibm.com

Chapter 1. What is RSCT?

RSCT (Reliable Scalable Cluster Technology) is a set of software components that together provide a comprehensive clustering environment for AIX and Linux. RSCT is the infrastructure used by a variety of IBM products to provide clusters with improved system availability, scalability, and ease of use. This chapter provides an overview of the RSCT components. It describes:

- **the Resource Monitoring and Control (RMC) subsystem.** This is the scalable, reliable backbone of RSCT. It runs on a single machine or on each node (operating system image) of a cluster and provides a common abstraction for the resources of the individual system or the cluster of nodes. You can use RMC for single system monitoring, or for monitoring nodes in a cluster. In a cluster, however, RMC provides global access to subsystems and resources throughout the cluster, thus providing a single monitoring/management infrastructure for clusters.
- **the RSCT core resource managers.** A resource manager is a software layer between a resource (a hardware or software entity that provides services to some other component) and RMC. A resource manager maps programmatic abstractions in RMC into the actual calls and commands of a resource.
- **the RSCT cluster security services,** which provide the security infrastructure that enables RSCT components to authenticate the identity of other parties.
- **the Topology Services subsystem,** which, on some cluster configurations, provides node/network failure detection.
- **the Group Services subsystem,** which, on some cluster configurations, provides cross node/process coordination.

What are management domains and peer domains?

In order to understand how the various RSCT components are used in a cluster, you should be aware that nodes of a cluster can be configured for either manageability or high availability.

You configure a set of nodes for manageability using the Clusters Systems Management (CSM) product as described in the *IBM Cluster Systems Management: Administration Guide*. The set of nodes configured for manageability is called a *management domain* of your cluster.

You configure a set of nodes for high availability using RSCT's Configuration resource manager. The set of nodes configured for high availability is called an RSCT *peer domain* of your cluster. For more information, refer to Chapter 3, "Creating and administering an RSCT peer domain," on page 19.

The following table lists the characteristics of the two domain types that can be present in your cluster.

A management domain:	A peer domain:
Established and administered by CSM.	Established and administered by RSCT's Configuration resource manager.

A management domain:	A peer domain:
Has a management server that is used to administer a number of managed nodes. Only management servers have knowledge of the whole domain. Managed nodes only know about the servers managing them. Managed nodes know nothing of each other.	Consists of a number of nodes with no distinguished or master node. All nodes are aware of all other nodes, and administration commands can be issued from any node in the domain. All nodes have a consistent view of the domain membership.
Processor architecture and operating system are heterogeneous.	Processor architecture and operating system are heterogeneous. Starting with RSCT version 2.3.2.0, peer domain nodes can run either AIX or Linux. AIX nodes will support any processor architecture supported by the AIX operating system. The supported Linux distributions are detailed in "Supported Linux distributions for RSCT 2.4.1.0" on page 16. (Please note, however, that products designed to run in a peer domain may not support the same heterogeneous environment as RSCT. Please refer to the specific exploiter's documentation for information on supported processor architecture and operating systems.)
The RMC subsystem and core resource managers are used by CSM to manage cluster resources. CSM also provides an additional resource manager — the Domain resource manager.	The RMC subsystem and core resource managers are used to manage cluster resources.
RSCT cluster security services are used to authenticate other parties.	RSCT cluster security services are used to authenticate other parties.
The Topology Services subsystem is not needed.	The Topology Services subsystem provides node/network failure detection.
The Group Services subsystem is not needed.	The Group Services subsystem provides cross node/process coordination.

| Although your cluster may be divided into management and peer domains, keep in
| mind that an individual node can participate in both domain types. However, within
| a management domain, a management server cannot belong to the same peer
| domain as any of the managed nodes.

What is RMC?

The Resource Monitoring and Control (RMC) subsystem is the scalable backbone of RSCT that provides a generalized framework for managing resources within a single system or a cluster. Its generalized framework is used by cluster management tools to monitor, query, modify, and control cluster resources. RMC provides a single monitoring/management infrastructure for both RSCT peer domains (where the infrastructure is used by the Configuration resource manager) and management domains (where the infrastructure is used by CSM). RMC can also be used on a single machine, enabling you to monitor/manage the resources of that machine. However, when a group of machines, each running RMC, are clustered together (into management domains/peer domains), the RMC framework allows a process on any node to perform an operation on one or more *resources* on any other node in the domain. A *resource* is the fundamental concept of the RMC architecture; it is an instance of a physical or logical entity that provides services to some other component of the system. Examples of resources include lv01 on node

10, ethernet device en0 on node 14, IP address 9.117.7.21, and so on. A set of resources that have similar characteristics (in terms of services provided, configuration parameters, and so on) is called a *resource class*.

The resources and resource class abstractions are defined by a *resource manager*. A *resource manager* is a process that maps resource and resource class abstractions into actual calls and commands for one or more specific types of resources. A resource manager runs as a stand-alone daemon, and contains definitions of all resource classes that the resource manager supports. These definitions include a descriptions of all attributes, actions, and other characteristics of a resource class. An RMC Access Control List (ACL) defines the access permissions that authorized users have for manipulating and grouping a resource class. For complete information on RMC, refer to Chapter 4, “Managing and monitoring resources using RMC and resource managers,” on page 61.

What are the RSCT resource managers?

RSCT provides a core set of resource managers for managing base resources on single systems and across clusters. Additional resource managers are provided by cluster licensed program products (such as CSM, which contains the Domain Management resource manager).

Some resource managers provide lower-level instrumentation and control of system resources. Others are essentially Management Applications implemented as resource managers.

The RSCT core resource managers are:

- the **Audit Log resource manager** which provides a system-wide facility for recording information about the system’s operation. This is particularly useful for tracking subsystems running in the background. A command-line interface to the resource manager enables you to list and remove records from an audit log.
- the **Configuration resource manager** which provides the ability to create and administer an RSCT peer domain. This is essentially a management application implemented as a resource manager. A command-line interface to this resource manager enables you to create a new peer domain, add nodes to the domain, list nodes in the domain, and so on. Refer to Chapter 3, “Creating and administering an RSCT peer domain,” on page 19 for more information.
- the **Event Response resource manager** which provides the ability to take actions in response to conditions occurring in the system. This is essentially a management application implemented as a resource manager. Using its command-line interface, you can define a condition to monitor. This condition is composed of an attribute to be monitored, and an expression that is evaluated periodically. You also define a response for the condition; the response is composed of zero or more actions and is run automatically when the condition occurs. For more information, refer to “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98.
- the **File System resource manager** manages file systems.
- the **Host resource manager** manages resources related to an individual machine.
- the **Sensor resource manager** which provides a means to create a single user-defined attribute to be monitored by the RMC subsystem. For more information, refer to “Creating event sensor commands for monitoring” on page 109.

- the **CIM resource manager** which enables you to use RMC to query system configuration through Common Information Model (CIM) classes. For more information, refer to “Querying CIM properties” on page 122.

For more information on RMC and the core resource managers, refer to Chapter 4, “Managing and monitoring resources using RMC and resource managers,” on page 61.

What are the cluster security services?

The cluster security services are used by RSCT applications and components to perform authentication within both management and peer domains. Authentication is the process by which a cluster software component, using cluster security services, determines the identity of one of its peers, clients, or an RSCT subcomponent. This determination is made in such a way that the cluster software component can be certain the identity is genuine and not forged by some other party trying to gain unwarranted access to the system. Be aware that authentication is different from authorization (the process of granting or denying resources based on some criteria). Authorization is handled by RMC and is discussed in “Managing user access to resources using RMC ACL files” on page 77.

Cluster Security Services uses **credential based authentication**. This type of authentication is used in client/server relationships and enables:

- a client process to present information that identifies the process in a manner that cannot be imitated to the server.
- the server process to correctly determine the authenticity of the information from the client.

Credential based authentication involves the use of a third party that both the client and the server trust. For this release, only Host Based Authentication is supported, but other security mechanisms may be supported in the future. In the case of Host Based Authentication, the trusted third party is the UNIX[®] operating system. This method of authentication is used between RSCT and its client applications (such as CSM), and also by the configuration resource manager during the creation and addition of nodes to an RSCT peer domain.

For more information on the cluster security services, refer to Chapter 6, “Understanding and administering cluster security services,” on page 147.

What are Topology Services?

The Topology Services subsystem is used within an RSCT peer domain to provide other RSCT applications and subsystems with network adapter status, node connectivity information, and a reliable messaging service. The Topology Services subsystem runs as a separate daemon process on each machine (node) in the peer domain. The adapter and node connectivity information is gathered by these instances of the subsystem forming a cooperation ring called a “heartbeat” ring. In this ring, each Topology Services’ daemon process sends a heartbeat message to one of its neighbors and expects to receive a heartbeat from another. In this system of heartbeat messages, each member monitors one of its neighbors. If the neighbor stops responding, the member that is monitoring it will send a message to a particular Topology Services daemon that has been designated as a Group Leader.

In addition to heartbeat messages, connectivity messages are sent around all heartbeat rings. Connectivity messages for each ring will forward its messages to

other rings, so that all nodes can construct a connectivity graph. This graph is used by the reliable messaging service to determine the route to use when delivering a message to a destination node.

For more information on Topology Services, refer to Chapter 7, “The Topology Services subsystem,” on page 167.

What are Group Services?

The Group Services subsystem is used within an RSCT peer domain to provide other RSCT applications and subsystems with a distributed coordination and synchronization service. The Group Services subsystem runs as a separate daemon process on each machine (node) in the peer domain. A group is a named collection of processes. Any process may create a new group, or join an existing group, and is considered a Group Services client. Group Services guarantees that all processes in a group see the same values for the group information, and that they see all changes to the group information in the same order. In addition, the processes may initiate changes to the group information.

A client process may be a *provider* or a *subscriber* of Group Services. *Providers* are full group members, and take part in all group operations. *Subscribers* merely monitor the group and are not able to initiate changes in the group.

For more information on Group Services, refer to Chapter 8, “The Group Services subsystem,” on page 185.

What are IBM Virtual shared disks and IBM Recoverable virtual shared disks?

IBM Virtual shared disks and IBM Recoverable virtual shared disks are subsystems of the AIX implementation of RSCT. These RSCT subsystems are provided as part of the AIX operation system and are not available in the Linux implementation of RSCT.

- IBM Virtual shared disk is an RSCT subsystem that lets application programs that are running on different nodes of an RSCT peer domain access a raw logical volume as if it were local at each of the nodes. Each virtual shared disk corresponds to a logical volume that is actually local at one of the nodes, which is called the *server node*. The Virtual shared disk subsystem routes I/O requests from the other nodes, called *client nodes*, to the server node and returns the results to the client nodes.

The I/O routing is done by the Virtual shared disk device driver that interacts with the AIX Logical Volume Manager (LVM). The device driver is loaded as a kernel extension on each node. Thus, raw logical volumes can be made globally accessible.

- The IBM Recoverable virtual shared disk (RVSD) is an RSCT subsystem that provides recoverability of your virtual shared disks if a node, adapter, or disk failure occurs. The RVSD subsystem manages your virtual shared disks, and, when an error is detected, will automatically switch disk access to an active node. Recovery is transparent to applications and there is no disruption of service except for a slight delay while takeover occurs.

IBM Virtual shared disks and IBM Recoverable virtual shared disks are implemented as an RMC Resource Manager (the Virtual Shared Disk Resource Manager) which provides a command line interface for configuring and managing virtual shared disks.

The IBM Virtual shared disks and IBM Recoverable virtual shared disks subsystems are not described in this book. For complete administrative information for these subsystems, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Managing Shared Disks* manual. For complete syntax information on the VSD commands, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference*.

What is the low-level application programming interface (LAPI)?

The low-level application programming interface (LAPI) is a component of the AIX implementation of RSCT. LAPI is provided as part of the AIX operation system and is not available in the Linux implementation of RSCT.

The low-level application programming interface (LAPI) is a message-passing API that provides a one-sided communication model. In this model, one task initiates a communication operation to a second task. The completion of the communication does not require the second task to take a complementary action. RSCT LAPI provides optimal communication performance on an IBM eServer™ pSeries High Performance Switch (pSeries HPS). PSSP LAPI provides optimal communication performance on the SP™ Switch and the SP Switch2.

The LAPI library provides basic operations to “put” data to and “get” data from one or more virtual addresses of a remote task. LAPI also provides an active message infrastructure. With active messaging, programmers can install a set of handlers that are called and run in the address space of a target task on behalf of the task originating the active message. Among their other uses, these handlers can be used to dynamically determine the target address (or addresses) where data from the originating task must be stored. You can use this generic interface to customize LAPI functions for your environment.

Some of LAPI’s other general characteristics include:

- Flow control
- Support for large messages
- Support for generic non-contiguous messages
- Non-blocking calls
- Interrupt and polling modes
- Efficient exploitation of switch functions
- Event monitoring support (to simulate blocking calls, for example) for various types of completion events

LAPI is meant to be used by programming libraries, and by power programmers for whom performance is more important than code portability.

LAPI is not described in this book. For complete conceptual, procedural, and reference information about this RSCT component, refer to the *Reliable Scalable Cluster Technology for AIX 5L: LAPI Programming Guide*

What is the System Resource Controller (SRC)?

The System Resource Controller (SRC) provides a set of commands and subroutines to make it easier for the system manager and programmer to create and control subsystems. A subsystem is any program or process or set of programs or processes that is usually capable of operating independently or with a controlling system. A subsystem is designed as a unit to provide a designated function. Specifically, the RSCT subsystems (Topology Services, Group Services, Cluster

Security Services, and so on) run under the SRC. On AIX, the SRC is, like the RSCT components, part of the operating system. For the Linux implementation of RSCT, the SRC is packaged with the RSCT components.

The SRC was designed to minimize the need for operator intervention. While it provides a consistent user interface for starting subsystems, stopping subsystems, and performing status inquiries, its operation should be largely transparent to you. Under normal circumstances, you should not explicitly start or stop the RSCT subsystems. However, when following certain troubleshooting procedures outlined in this book, you may be instructed to use the SRC commands **startsrc** and **stopsrc** to start and stop RSCT subsystems. You can also use the command **lssrc** to list the status of RSCT systems.

Chapter 2. RSCT installation and software verification

The AIX implementation of RSCT is included as part of the AIX 5L operating system. The Linux implementation of RSCT is included with a variety of Linux-based products such as IBM Cluster System Management (CSM) for Linux and IBM Tivoli System Automation for Multiplatforms.

This book applies to:

- RSCT version 2.3.5.0 for AIX 5L (version 5.2).
- RSCT version 2.4.1.0 for AIX 5L (version 5.3) and Linux.

A new fileset, `rsct.core.lprm` (Least Privilege Resource Manager), has been introduced in RSCT 2.3.5.0. This fileset will not be installed automatically with RSCT 2.3.5 PTF updates. The `rsct.core.lprm` fileset may be explicitly installed when RSCT 2.3.5 PTF updates are applied. If it is, be aware that, in order to reject the updates, you will need to uninstall the fileset.

RSCT installation verification on AIX nodes

To verify that RSCT has been installed on an AIX node, enter:

```
lspp -L rsct.*
```

Output should be similar to the following.

Fileset	Level	State	Type	Description (Uninstaller)
<code>rsct.basic.hacmp</code>	2.4.1.0	C	F	RSCT Basic Function (HACMP/ES Support)
<code>rsct.basic.rte</code>	2.4.1.0	C	F	RSCT Basic Function
<code>rsct.basic.sp</code>	2.4.1.0	C	F	RSCT Basic Function (PSSP Support)
<code>rsct.compat.basic.hacmp</code>	2.4.1.0	C	F	RSCT Event Management Basic Function (HACMP/ES Support)
<code>rsct.compat.basic.rte</code>	2.4.1.0	C	F	RSCT Event Management Basic Function
<code>rsct.compat.basic.sp</code>	2.4.1.0	C	F	RSCT Event Management Basic Function (PSSP Support)
<code>rsct.compat.clients.hacmp</code>	2.4.1.0	C	F	RSCT Event Management Client Function (HACMP/ES Support)
<code>rsct.compat.clients.rte</code>	2.4.1.0	C	F	RSCT Event Management Client Function
<code>rsct.compat.clients.sp</code>	2.4.1.0	C	F	RSCT Event Management Client Function (PSSP Support)
<code>rsct.core.auditrm</code>	2.4.1.0	C	F	RSCT Audit Log Resource Manager
<code>rsct.core.erm</code>	2.4.1.0	C	F	RSCT Event Response Resource Manager
<code>rsct.core.fsrn</code>	2.4.1.0	C	F	RSCT File System Resource Manager
<code>rsct.core.gui</code>	2.4.1.0	C	F	RSCT Graphical User Interface
<code>rsct.core.hostrm</code>	2.4.1.0	C	F	RSCT Host Resource Manager
<code>rsct.core.lprm</code>	2.4.1.0	C	F	RSCT Least Privilege Resource Manager
<code>rsct.core.rmc</code>	2.4.1.0	C	F	RSCT Resource Monitoring and Control
<code>rsct.core.sec</code>	2.4.1.0	C	F	RSCT Security
<code>rsct.core.sensorrm</code>	2.4.1.0	C	F	RSCT Sensor Resource Manager
<code>rsct.core.sr</code>	2.4.1.0	C	F	RSCT Registry
<code>rsct.core.utils</code>	2.4.1.0	C	F	RSCT Utilities
<code>rsct.exp.cimrm</code>	2.4.1.0	C	F	CIM Resource Manager

If the RSCT components are installed, you'll want to make sure that they are the version that apply to this book. This book applies to RSCT version 2.3.5.0 and 2.4.1.0. If you are using RSCT specifically in conjunction with an exploiter of the technology (such as CSM for AIX 5L or LoadLeveler® for AIX 5L), you'll want to make sure that this is the version of RSCT required by the exploiter. You should also be aware that not all of the RSCT filesets are required by every RSCT exploiter. Refer to the specific RSCT exploiter's documentation for information on RSCT version and fileset requirements.

If you discover you need a later version of this or other RSCT documentation, refer to the **IBM eServer Cluster Information Center**. This web site is located at <http://publib.boulder.ibm.com/clresctr> and always contains the most recent RSCT documentation in PDF and HTML formats. Both the current RSCT books and earlier versions of the library are also available in PDF format from the **IBM Publications Center** Web site located at <http://www.ibm.com/shop/publications/order>.

The RSCT for AIX 5L filesets are described in the following table.

Table 1. RSCT for AIX 5L Filesets

This Fileset:	Contains:
rsct.basic.rte	configuration resource manager group services topology services
rsct.core	Core RSCT components including the: <ul style="list-style-type: none"> • audit log resource manager • event response resource manager (ERRM) • file system resource manager • host resource manager • resource monitoring and control (RMC) subsystem • cluster security services • sensor resource manager • system registry • miscellaneous utilities
rsct.exp	the RSCT Expansion Pack. The RSCT Expansion Pack contains the CIM resource manager. This fileset is part of the AIX 5L Expansion Pack and will not be installed by default. To use the CIM resource manager on AIX, you will also need to install the Pegasus CIM Server filesets off the AIX 5L Expansion Pack. The filesets for the Pegasus CIM server are: sysmgt.pegasus.cimserver sysmgt.pagasus.osbaseproviders
rsct.vsd	the virtual shared disk and recoverable virtual shared disk subsystems
rsct.lapi	the low-level application programming interface (LAPI)

If entering the **lspp** command as described above reveals that needed RSCT filesets are not installed, you can install them from the AIX installation media using the **installp** command. Enter the **installp** command as shown below, where *cd0* is the name of the AIX installation media, and *fileset* is the name of an RSCT fileset as shown in the preceding table.

```
installp -agXd /dev/cd0 fileset
```

RSCT installation verification on Linux nodes

The Linux implementation of RSCT is shipped with a number of products that exploit the technology. The RSCT filesets should be installed by following the specific exploiter's installation procedure. Before installing RSCT, you should make sure that the target node has the required packages shown in the following table:

Library		Package
Standard C Library		glibc
Standard C++ Library		libstdc++
Compatibility standard C++ Library	On Red Hat:	compat-libstdc++
	On SLES 8	compat

To verify that RSCT has been installed on a Linux node, enter:

```
rpm -qa | grep -E -e "rsct|src"
```

Output should be similar to:

```
src-1.2.1.1-0
rsct.core.utils-2.4.1.0-0
rsct.basic-2.4.1.0-0
rsct.core-2.4.1.0-0
```

If your system has a 64-bit kernel and the IBM General Parallel File System (GPFS) version 2.2 or earlier installed, the output should be similar to:

```
src-1.2.1.1-0
rsct.core.utils-2.4.1.0-0
rsct.basic-2.4.1.0-0
rsct.core-2.4.1.0-0
rsct.64bit-2.4.1.0-0
```

If the RSCT components are installed, check to make sure that they are at the version level that applies to this book. This book applies to RSCT for Linux version 2.4.1.0. If you discover you need a later version of this or other RSCT documentation, refer to the **IBM eServer Cluster Information Center**. This web site is located at <http://publib.boulder.ibm.com/clresctr> and always contains the most recent RSCT documentation in PDF and HTML formats. Both the current RSCT books and earlier versions of the library are also available in PDF format from the **IBM Publications Center** Web site located at <http://www.ibm.com/shop/publications/order>.

The following table describes the RSCT for Linux RPM packages. In the RPM package names shown in this table, the *platform* will be i386, ppc, s390, ppc64, or x86_64.

Table 2. RSCT for Linux RPM Packages

This RPM Package:	Contains:
rsct.basic-2.4.1.0-0. <i>platform</i> .rpm	configuration resource manager group services topology services

Table 2. RSCT for Linux RPM Packages (continued)

This RPM Package:	Contains:
rsct.core-2.4.1.0-0. <i>platform</i> .rpm	resource monitoring and control (RMC) audit log resource manager event response resource manager (ERRM) file system resource manager host resource manager cluster security services system registry
rsct.core.cimrm-2.4.1.0. <i>platform</i> .rpm	CIM resource manager (where available)
rsct.core.utils-2.4.1.0-0. <i>platform</i> .rpm	miscellaneous utilities
src-1.2.1.1-0. <i>platform</i> .rpm	system resource controller
rsct.64bit.SLES-2.4.1.0-0. <i>platform</i> .rpm	group services library for SUSE Linux Enterprise Server 8 (SLES 8) 64-bit kernel
rsct.64bit.RH-2.4.1.0-0. <i>platform</i> .rpm	group services library for Red Hat 64-bit kernel

If entering the **rpm** command as described above reveals that needed RSCT RPM packages are not installed, you can install them from the RSCT exploiter's installation media. You should refer to the RSCT exploiter's documentation for installation instructions.

You can install RSCT by itself, but, due to dependencies among the RPM packages, the packages must be installed in a specific sequence (as shown in the following instructions). In the following instructions, replace *platform* with *i386*, *ppc*, *s390*, *ppc64*, or *x86_64* as appropriate for your system platform.

1. Install the system resource controller by entering:
`rpm -i src-1.2.1.1-0.platform.rpm`
2. Install the RSCT utilities by entering:
`rpm -i rsct.core.utils-2.4.1.0-0.platform.rpm`
3. Install the RSCT core components by entering:
`rpm -i rsct.core-2.4.1.0-0.platform.rpm`
4. Install the RSCT basic components by entering:
`rpm -i rsct.basic-2.4.1.0-0.platform.rpm`
5. If your system has a 64-bit kernel, and you plan to install GPFS (version 2.2 or earlier), you will need to install an additional 64-bit RPM package. If you are using the AMD64 platform, enter:
`rpm -i rsct.64bit.SLES-2.4.1.0-0.x86_64.rpm`

Despite the name of the RPM package, it can be used for Red Hat distributions as well as SLES provided you are using the AMD64 platform. If you are not using the AMD64 platform, refer to the following table for more information on the package you need to install.

If you are using:	Enter:
Red Hat Linux	<code>rpm -i rsct.64bit.RH-2.4.1.0-0.<i>platform</i>.rpm</code>
SUSE LINUX Enterprise Server 8 (SLES 8)	<code>rpm -i rsct.64bit.SLES-2.4.1.0-0.<i>platform</i>.rpm</code>

If entering the **rpm** command as described in the preceding instructions reveals that previous versions of RSCT RPM packages are installed, you could upgrade RSCT using the **rpm** command.

If:		You can upgrade RSCT using the following command:
Your system does not have the rsct64bit package installed		<code>rpm -Fvh src-1.2.1.1-0.platform.rpm rsct.core.utils-2.4.1.0-0.platform.rpm rsct.core-2.4.1.0-0.platform.rpm rsct.basic-2.4.1.0-0.platform.rpm</code>
Has the rsct64bit package installed and is running...	Red Hat Linux	<code>rpm -Fvh src-1.2.1.1-0.platform.rpm rsct.core.utils-2.4.1.0-0.platform.rpm rsct.core-2.4.1.0-0.platform.rpm rsct.basic-2.4.1.0-0.platform.rpm rsct.64bit.RH-2.4.1.0-0.platform.rpm</code>
	SLES 8	<code>rpm -Fvh src-1.2.1.1-0.platform.rpm rsct.core.utils-2.4.1.0-0.platform.rpm rsct.core-2.4.1.0-0.platform.rpm rsct.basic-2.4.1.0-0.platform.rpm rsct.64bit.SLES-2.4.1.0-0.platform.rpm</code>

If your system has any RSCT-exploiter packages installed, you may have to upgrade those RPM packages as well. You should refer to the RSCT exploiter's documentation for appropriate instructions.

If you wish to uninstall RSCT, please note that the packages must be uninstalled in a specific sequence (as shown in the following instructions). If there is any exploiter that has dependency on RSCT, the **rpm** command will not allow you to uninstall the RSCT packages.

1. If the rsct64bit package was installed, uninstall it by entering:

```
rpm -e rsct.64bit
```

2. Uninstall the RSCT basic components by entering:

```
rpm -e rsct.basic
```

3. Uninstall the RSCT core components by entering:

```
rpm -e rsct.core
```

4. Uninstall the RSCT utilities by entering:

```
rpm -e rsct.core.utils
```

5. Uninstall the system resource controller by entering:

```
rpm -e src
```

The Linux distributions supported by this version of RSCT are described next in "Supported Linux distributions for RSCT 2.4.1.0" on page 16. Please check your RSCT exploiter's documentation to see if that particular product also supports a particular distribution.

If you are installing RSCT on:

- Red Hat AS 2.1 on x86, refer to the additional instructions in "Required RSCT fix for Red Hat AS 2.1 on x86" on page 14.
- Red Hat 8.0 on x86, refer to the additional instructions in "Applying required patch for Red Hat 8.0 on x86" on page 14.
- Red Hat 9.0 on x86, refer to the additional instructions in "Applying required patch for Red Hat 9.0 on x86" on page 15.
- Red Hat EL 3.0 on AMD-64, refer to the additional instructions in "Kernel requirement for Red Hat EL 3.0 on AMD-64" on page 15.

- SUSE Linux Enterprise Server 8 (SLES 8) on pSeries, refer to the additional instructions in “Kernel requirement for SUSE SLES 8 Linux on pSeries” on page 15.
- SUSE Linux Enterprise Server 8 (SLES 8) on iSeries™, refer to the additional instructions in “Kernel requirement for SUSE SLES 8 Linux on iSeries” on page 15.
- Red Hat EL 3 on Power, refer to the additional instructions in “Obtaining compat-libstdc++ for Red Hat EL 3 on Power” on page 15.
- SUSE Linux Enterprise Server 9 (SLES 9) refer to the additional instructions in “Kernel update requirement for SUSE SLES 9 Linux on all platform” on page 15.

Required RSCT fix for Red Hat AS 2.1 on x86

Versions 2.96-116 and 2.96-118 of the libstdc++ libraries are not compatible with RSCT. Because the version of the libstdc++ libraries that comes with the Red Hat AS 2.1 Update1 and Update2 CDs are not compatible with RSCT, you must apply version 2.96-124 of libstdc++ before you install RSCT.

If your system has Red Hat AS 2.1 Update1 or Update2 installed, perform the following steps before you install RSCT:

1. Go to the following website:

<http://rhn.redhat.com/errata/RHBA-2003-358.html>

and download the libstdc++-2.96-124 RPM package, and, optionally, the libstdc++-devel-2.96-124 RPM package.

2. Use the following command to update the libstdc++ package with the downloaded version:

```
rpm -Fvh libstdc++
```

3. Install RSCT.

Applying required patch for Red Hat 8.0 on x86

Red Hat updated the glibc package to fix pthread stack overflow. If not applied, certain RSCT subsystems could crash. After installation of Red Hat 8.0, but before you install RSCT, perform the following steps:

1. Use the following command to check glibc level on your system:

```
rpm -qa | grep glibc
```

2. If your system doesn't have glibc-2.3.2-4-80.6 or later version, go to the following website:

<https://rhn.redhat.com/errata/RHSA-2003-089.html>

and download the following required RPM package:

```
glibc-2.3.2-4.80.6.i686.rpm
```

and, if desired, any of these optional RPM packages:

```
glibc-common-2.3.2-4.80.6.i386.rpm
```

```
glibc-debug-2.3.2-4.80.6.i686.rpm
```

```
glibc-devel-2.3.2-4.80.6.i386.rpm
```

3. Use the following command to update the glibc package with the downloaded version:

```
rpm -Fvh glibc*
```

4. Install RSCT.

Applying required patch for Red Hat 9.0 on x86

Red Hat updated the glibc package to fix a number of bugs. If not applied, certain RSCT subsystems could hang or crash. After installation of Red Hat 9.0, but before you install RSCT, perform the following steps:

1. Use the following command to check glibc level on your system:

```
rpm -qa | grep glibc
```

2. If your system doesn't have glibc-2.3.2-27-9 or later version, go to the following website:

<https://rhn.redhat.com/errata/RHSA-2003-325.html>

and download the following required RPM package:

```
glibc-2.3.2-27.9.7.i686.rpm
```

and, if desired, any of these optional RPM packages:

```
glibc-common-2.3.2-27.9.7.i386.rpm  
glibc-debug-2.3.2-27.9.7.i386.rpm  
glibc-devel-2.3.2-27.9.7.i386.rpm  
glibc-profile-2.3.2-27.9.7.i386.rpm  
glibc-utils-2.3.2-27.9.7.i386.rpm  
nptl-devel-2.3.2-27.9.7.i686.rpm  
nscd-2.3.2-27.9.7.i386.rpm
```

3. Use the following command to update the glibc package with the downloaded version:

```
rpm -Fvh glibc* nptl* nscd*
```

4. Install RSCT.

Kernel requirement for Red Hat EL 3.0 on AMD-64

On Red Hat EL 3 on AMD-64, RSCT requires Update 2. Without Red HAT EL 3 Update 2 installed, RSCT may be unstable in cluster mode.

Kernel requirement for SUSE SLES 8 Linux on pSeries

On SUSE LINUX Enterprise Server 8 (SLES 8) on pSeries, RSCT requires Service Pack 3 (SP3). Without SP3 installed, the IBM.Host resource class may be unable to correctly report values.

Kernel requirement for SUSE SLES 8 Linux on iSeries

On SUSE LINUX Enterprise Server 8 (SLES 8) on iSeries, RSCT requires Service Pack 3 (SP3). Without SP3 installed, RSCT may be unable to properly handle devices or system resources after system reboot.

Obtaining compat-libstdc++ for Red Hat EL 3 on Power

Your Red Hat EL 3 on Power installation CD may not include the compat-libstdc++ rpm package. Without the compat-libstdc++ package, you cannot install RSCT . If you can't locate compat-libstdc++ rpm from your installation CD, go to the following web site:

<https://rhn.redhat.com/network/software/packages/details.pxt?pid=199449>

and download the compat-libstdc++-7.3-2.96.123.ppc.rpm package.

Kernel update requirement for SUSE SLES 9 Linux on all platform

On SUSE Linux Enterprise Server 9 (SLES 9), RSCT requires SLES 9 updates if your system:

- has SLES 9 GA-level installed,
- the compat service is in effect for the group database,
- and the **/etc/groups** file directs the use of NIS.

On a SLES 9 system, the lookup order of the group database is controlled by the Name Service Switch (NSS). The NSS configuration file may specify the compat service for the group database. When the compat service is specified for the group database, lookup requests are processed by searching the local **/etc/groups** file, and special entries within the **/etc/groups** file can direct the use of the Network Information Service (NIS).

The Name Service Switch (NSS) configuration file is **/etc/nsswitch.conf**. The compat service may be specified explicitly for the group database in this file. If the file does not exist, or does not contain an entry for the group database, the compat service is used by default for group database lookups. See the **nsswitch.conf(5)** man page for details. When the compat service is in effect for group database lookups, the following entry in **/etc/groups** directs the use of Network Information Service (NIS).

```
+::::
```

The SLES 9 updates are required because program exceptions may otherwise occur when routines in the C library are called to search the group database while the compat service is in effect for the group database and the **/etc/groups** file directs the use of NIS. Because program exceptions can occur if SLES 9 updates are not installed, some RSCT daemons will not function properly.

Supported Linux distributions for RSCT 2.4.1.0

The following table lists the Linux distributions supported by RSCT 2.4.1.0. Please note that Red Hat EL 2.1 is no longer supported.

Table 3. Supported Linux distributions for RSCT 2.4.1.0

Linux Distribution	Hardware						
	xSeries®			pSeries		zSeries®	iSeries
	x86	AMD-64	xBlade	Power 4	JS20 Blade		
Red Hat 7.2	supports 32-bit distribution	not supported	not supported	not supported	not supported	not supported	not supported
Red Hat 7.3	supports 32-bit distribution	not supported	supports 32-bit distribution	not supported	not supported	not supported	not supported
Red Hat 8.0	supports 32-bit distribution	not supported	not supported	not supported	not supported	not supported	not supported
Red Hat 9.0	supports 32-bit distribution	supports 32-bit distribution	not supported	not supported	not supported	not supported	not supported
Red Hat AS 2.1	supports 32-bit distribution	not supported	supports 32-bit distribution	not supported	not supported	not supported	not supported
Red Hat EL 3.0 (support indicated includes all three members of the Red Hat EL family — AS, WS, and ES)	supports 32-bit distribution	supports 32-bit and 64-bit distribution	supports 32-bit distribution	supports 32-bit and 64-bit distribution	supports 32-bit and 64-bit distribution	supports 31-bit and 64-bit distribution	not supported

Table 3. Supported Linux distributions for RSCT 2.4.1.0 (continued)

SUSE LINUX 8.0	supports 32-bit distribution	not supported	not supported	not supported	not supported	not supported	not supported
SUSE LINUX 8.1	supports 32-bit distribution	not supported	supports 32-bit distribution	not supported	not supported	not supported	not supported
SuSE SLES 7	supports 32-bit distribution	not supported	not supported	not supported	not supported	supports the 31-bit and 64-bit distribution	not supported
SUSE LINUX Enterprise Server 8 (SLES 8)	supports 32-bit distribution	supports the 32-bit and 64-bit distribution	supports 32-bit distribution	supports the 32-bit and 64-bit distribution	supports the 32-bit and 64-bit distribution	supports the 31-bit and 64-bit distribution	supports the 64-bit distribution
SUSE LINUX Enterprise Server 9 (SLES 9)	supports the 32-bit distribution	supports the 64-bit distribution	supports the 32-bit distribution	supports the 64-bit distribution	supports the 64-bit distribution	not supported	not supported
Turbo SLES 8 (United Linux 1.0 only)	not supported	not supported	not supported	supports the 32-bit and 64-bit distribution	not supported	not supported	not supported
Connectiva LE Edition 8 (United Linux 1.0 only)	not supported	not supported	not supported	supports the 32-bit and 64-bit distribution	not supported	not supported	not supported

Chapter 3. Creating and administering an RSCT peer domain

This chapter describes how to use the configuration resource manager commands to create and administer an RSCT peer domain.

What is an RSCT peer domain?

An RSCT peer domain is a cluster of nodes configured for high availability. The peer domain could consist of all nodes in your cluster, or could be a subset of nodes in your overall cluster solution (which could also consist of nodes configured by CSM into a management domain). Within a management domain, however, the management server cannot belong to the same peer domain as any of the managed nodes.

An RSCT peer domain uses:

- RSCT cluster security services for authentication. (Refer to Chapter 6, “Understanding and administering cluster security services,” on page 147 for more information.)
- the Topology Services subsystem for node/network failure detection. Generally, the peer domain’s use of this subsystem will be transparent to you. (Refer to Chapter 7, “The Topology Services subsystem,” on page 167 for more information.)
- the Group Services subsystem for cross node/process coordination. Generally, the peer domain’s use of this subsystem will be transparent to you. (Refer to Chapter 8, “The Group Services subsystem,” on page 185 for more information.)
- the Resource Monitoring and Control subsystem for coordination between the various RSCT subsystems. Generally, the peer domain’s use of this subsystem will be transparent to you. However, you can use RMC to monitor the peer domain. (Refer to Chapter 4, “Managing and monitoring resources using RMC and resource managers,” on page 61 for more information.)

What is the configuration resource manager?

The configuration resource manager provides the ability to create and administer an RSCT peer domain. This is essentially a management application implemented as an RMC resource manager. A command-line interface to this resource manager enables you to create a new peer domain, add nodes to the domain, list nodes in the domain, and so on. Refer to “What can I do using configuration resource manager commands?” on page 22 for more information.

What are communication groups?

Communication groups control how liveness checks (in other words, Topology Services’ “heartbeats”) are performed between the communication resources within the peer domain. Each communication group corresponds to a Topology Services’ heartbeat ring, and identifies the attributes that control the liveness checks between the set of network interfaces and other devices in the group.

The configuration resource manager automatically forms communication groups when a new peer domain is formed. When you then bring a peer domain online, the configuration resource manager will supply the communication group definition to Topology Services. Topology Services will create the actual heartbeat rings needed to perform liveness checks for the peer domain nodes.

Each communication group has several characteristics. These characteristics specify:

- the number of missed heartbeats that constitute a failure
- the number of seconds between the heartbeats
- whether or not broadcast should be used
- whether or not source routing should be used

Each communication group also has a list of its member network interfaces.

The configuration resource manager may also form new communication groups as new nodes are added to the peer domain. When these added nodes are brought online in the peer domain, the configuration resource manager supplies the modified information to Topology Services. Topology Services may then modify existing heartbeat rings or create additional heartbeat rings.

In general, communication groups will be transparent to you. The configuration resource manager forms them in conjunction with the Topology Services subsystem as you issue commands to create and modify a peer domain. Although the configuration resource manager allows you to create your own communication groups, such manual configuration is neither necessary or advisable.

For more information, refer to “Understanding and working with communication groups” on page 39.

What is quorum?

Quorum refers to the minimum numbers of nodes within the peer domain that are required to carry out a particular operation. There are three kinds of quorum that specify the number of nodes required for different types of operations. These are *startup quorum*, *configuration quorum*, and *operational quorum*.

What is startup quorum?

Startup quorum refers to the number of nodes needed to bring a peer domain online. If the configuration resource manager is unable to reach this minimum number of nodes, it will not be able to start the peer domain.

What is configuration quorum?

Configuration quorum refers to the minimum number of nodes, or a certain peer-domain state, needed to perform operations that modify the peer domain's configuration information. If you issue a command that will modify a peer domain's configuration, and the configuration resource manager is unable to reach this minimum number of nodes, the command will fail.

What is operational quorum?

Operation quorum refers to the minimum number of nodes, or a certain peer-domain state, needed to safely activate resources without creating conflicts with another subdomain. It is used to protect data following domain partitioning.

What is domain partitioning?: Domain partitioning is when a peer domain is inadvertently divided into two or more sub-domains.

How does operational quorum help the configuration resource manager protect data following domain partitioning?: Following domain partitioning when critical resources are active on nodes, the configuration resource manager needs to determine which sub-domain can continue operating and which other(s) should be dissolved. This is especially important when there are applications running on the

domain that employ shared resource access. If the peer domain is partitioned, nodes in one sub-domain are no longer aware of nodes in any other sub-domain. Data corruption can occur if nodes in different sub-domains try to access the same shared resource. The configuration resource manager prevents this situation by deciding which sub-domain has operational quorum and can continue operating, thus becoming the peer domain. Usually, the sub-domain with the majority of nodes will have operational quorum.

What is a tie breaker?: After domain partitioning, it is usually the sub-domain with the majority of nodes will have operational quorum. However, sometimes there is a tie in which multiple sub-domains have exactly half of the defined nodes. A “tie” situation also occurs when exactly half the nodes of a domain are online, and the other half are inaccessible. When there is a tie, the configuration resource manager uses a *tie breaker* to determine which sub-domain has operational quorum. A *tie breaker* is an RMC resource defined by the configuration resource manager that specifies how tie situations should be resolve. It is the tie-breaker that determines which sub-domain will have operational quorum and so will survive, and which sub-domain will be dissolved.

For more information, refer to “Determining how the configuration resource manager will resolve tie situations when calculating operational quorum” on page 52.

What is a critical resource protection method?: When a sub-domain that has critical resources loses quorum, the configuration resource manager uses a *critical resource protection method* on each node of the sub-domain to ensure that critical resources will not be corrupted. A *critical resource protection method* is simply software that determine how the configuration resource manger will respond when quorum is lost in a sub-domain. A critical resource protection method will also be used on a node whose configuration resource manager, group services, or topology services daemon hangs. There are a number of critical resource protection methods defined by the configuration resource manager. You can specify a critical resource protection method for the entire peer domain, or specify one to be used on just one particular node. The critical resource protection methods do such things as halt the system, reset and reboot the system, and so on.

For more information, refer to “Setting the critical resource protection method for a peer domain or a node in a peer domain” on page 49.

What are quorum types?

A peer domain’s quorum type specifies how startup quorum, configuration quorum, and operational quorum will be calculated for the peer domain. The quorum types are:

Normal

Normal mode which is the default for an AIX/Linux cluster. In this mode:

StartupQuorum = $N/2$
ConfigQuorum = $N/2 + 1$
OpQuorum = Majority + TieBreaker

Quick Quick startup mode, which is useful for large clusters. In this mode:

StartupQuorum = 1
ConfigQuorum = $N/2 + 1$
OpQuorum = Majority + TieBreaker

Override

Override mode. Available only for OS/400 environments, and the default for such environments. In this mode:

StartupQuorum = 1

ConfigQuorum = 1

OpQuorum is externally provided by RMC exploiter.

SANFS

SANFS mode. Available only for environments with the IBM TotalStorage SAN File System, and the default for such environments. In this mode:

StartupQuorum = 1

ConfigQuorum is externally provided by a designated group state value.

OpQuorum = Majority + TieBreaker

What can I do using configuration resource manager commands?

The following table outlines the tasks you can perform using configuration resource manager commands.

To:	Use:	For more information, refer to:
Create a peer domain	<ol style="list-style-type: none">1. The preprnode command to prepare the security environment on each node that will participate in the peer domain.2. The mkrpdomain command to create a new peer domain definition.3. The startdomain command to bring the peer domain online.	"Creating a peer domain" on page 25
Add nodes to an existing peer domain	<ol style="list-style-type: none">1. The preprnode command to prepare the security environment on the new node.2. The addrpnode command to add the node to a peer domain.3. The startpnode command to bring the node online.	"Adding nodes to an existing peer domain" on page 32
Take a peer domain node offline	The stoprnode command	"Taking a peer domain node offline" on page 36
Take a peer domain offline	The stopdomain command	"Taking a peer domain offline" on page 37
Remove a node from a peer domain	The rmrnode command	"Removing a node from a peer domain" on page 37
Remove a peer domain	The rmrdomain command	"Removing a peer domain" on page 38
List communication groups. Communication groups control how liveness checks (Topology Services' "heartbeats") are performed between the communication resources within the peer domain.	The lscomg command	"Listing communication groups" on page 40

To:	Use:	For more information, refer to:
Modify a communication group's characteristics (Topology Services' tunables)	<p>the chcomg command to</p> <ul style="list-style-type: none"> • specify the communication group's sensitivity setting (the number of missed heartbeats that constitute a failure). • specify the communication group's period setting (the number of seconds between heartbeats). • specify the communication group's priority setting (the importance of this communication group with respect to others). • specify the communication group's broadcast setting (whether or not to broadcast if the underlying network supports it). • specify the communication group's source routing setting (in case of adapter failure, whether or not source routing should be used if the the underlying network supports it). 	"Modifying a communication group's characteristics" on page 41
Manually configure communication groups (not necessary under normal circumstances; only to be exercised in unavoidable situations)	the chcomg command to modify a communication group's network interface.	"Modifying a communication group's network interface" on page 43
	the mkcomg command to create a communication group.	"Creating a communication group" on page 44
	the rmcomg command to remove a communication group.	"Removing a communication group" on page 46

In addition to the tasks you can perform using configuration resource manager commands, this chapter also describes how you can use generic RMC commands to:

- modify topology services and group services parameters, and
- determine how the configuration manager responds to domain partitioning to prevent corruption of critical data.

For more information see "Modifying Topology Services and Group Services parameters" on page 46 and "Determining how your system responds to domain partitioning and subsystem daemon failure" on page 48.

When describing how to perform these administrative tasks, this chapter shows command examples, but does not necessarily contain a description of all of the command options. For complete syntax of any of the commands described in this chapter, refer, depending on the operating system, to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*. If you encounter error messages while trying to perform the tasks outlined in the chapter, refer to the manual *Reliable Scalable Cluster Technology: Messages* for recovery information.

Prerequisites and restrictions to using configuration resource manager commands

Before using configuration resource manager commands to perform the tasks described in this chapter, you should be aware of the following prerequisites and restrictions.

- The following packages are required. On AIX, these are available as part of the base AIX operating system. On Linux, these packages are shipped with the products (such as CSM) that use the RSCT technology, and should have been installed as part of the product's installation procedure.
 - rsct.core
 - rsct.basic

- rsct.core.utils
- rsct.core.sec (required for AIX nodes only)
- All nodes you plan to include in the peer domain must be reachable from all other nodes. While you can have multiple networks and routers to accomplish this, there must be IP connectivity between all nodes of the peer domain.

Supported RSCT versions

RSCT Peer Domain is officially supported by RSCT with a version number of 2.2.1.20 or higher. Although it was possible to create an RSCT Peer Domain with an earlier version (RSCT 2.2.1.10), that version is not officially supported. Nodes running RSCT 2.2.1.10 should **not** be added to an Peer Domain created with RSCT 2.2.1.20 or a later version.

To verify the RSCT version installed on an AIX node, enter the command:

```
lslpp -l rsct*
```

To verify the RSCT version installed on a Linux node, enter the command:

```
rpm -qa | grep rsct
```

RSCT Support for VIPA

As of RSCT releases 2.3.5 and 2.4.1, the AIX implementation of RSCT allows you to configure Virtual IP Address (VIPA) interfaces on nodes that are part of a peer domain or a management domain. Once VIPA is configured to include a set of physical network interfaces, outgoing packets that go through one of these interfaces will have the VIPA interface address as source address. As a result, the destination of the packet, and intervening routers, needs to have a route that allows it to communicate back with the VIPA address. These routes are needed even if there is no application on the nodes that communicates using the VIPA address. Failure to create these routes may result in nodes that fail to communicate through TCP or UDP — even though the **ping** command may still show connectivity to be intact.

Unlike other network interfaces, VIPA interfaces are not monitored using Topology Services' heartbeating. The output of the **lsrsrc IBM.NetworkInterface** command will reveal that the HeartbeatActive attribute is set to 0 for all VIPA interfaces. Manually attempting to set the attribute to 1 will have no effect.

For details on how to configure VIPA interfaces, refer to *AIX 5L System Management Guide: Communications and Networks*, SC23-4909.

For information on whether VIPA is supported by a particular RSCT exploiter, refer to the exploiter's product documentation. Special attention is required regarding which of the interfaces on a node are allowed to be part of the VIPA, and a particular RSCT exploiter might require that given networks should not be made part of VIPA.

Migration

Avoiding Domain Partitioning When Migrating From RSCT 2.2.1.x or 2.3.0.x

AIX 5.1 nodes running with the RSCT level 2.2.1.x , or AIX 5.2 nodes running with the RSCT level 2.3.0.x, cannot be migrated to RSCT version 2.3.3.0 while online in a peer domain that contains nodes running with a

level of RSCT 2.3.1.x or higher. If nodes running RSCT 2.2.1.x or 2.3.0.x are migrated while online in a peer domain containing RSCT 2.3.1.x or higher nodes, a partitioned peer domain may be created when the migration completes.

Before migrating an individual node running RSCT 2.2.1.x or 2.3.0.x, take the node offline using the **stoprnode** command (as described in “Taking a peer domain node offline” on page 36). After the node completes migration, you can restart it using the **startprnode** command (as described in “Step 3: bring node online in the peer domain” on page 35).

If the peer domain is partitioned, you can fix this problem by stopping all nodes in both sides of the partition and then restarting the peer domain (using the **startprdomain** command as described in “Step 3: bring the peer domain online” on page 29) from a node running the higher level of RSCT.

PTF Rejection Can Result in Loss of Cluster Data

When a node is upgraded from a version prior to RSCT 2.3.3.0, the registry information will be saved and converted to a new registry format. If an upgraded node is downgraded with PTF rejection, the new registry will be replaced by the old saved registry. If this happens, any cluster data committed after the upgrade will be lost.

In order to complete the migration of a peer domain and update the active RSCT version to a new level, you must enter the **runact** command as shown below. This command should be run only after all the nodes defined in a peer domain are upgraded to a later version. The command only needs to be run once on one of the online nodes with more than half of the nodes online. If all the upgraded nodes have an RSCT version higher than the active version (RSCTActiveVersion), the new minimum RSCT version across all nodes is determined and becomes the new active version of the peer domain.

To complete the migration of a peer domain:

1. Upgrade nodes defined in a peer domain to a later version.
2. After you have upgraded all the nodes defined in a peer domain, make sure more than half of the nodes are online. If not, then bring nodes online to meet the criteria.
3. Execute the following commands on one of the online nodes in the peer domain:
 - a. Set the management scope to RSCT Peer Domain (a value of 2):

```
export CT_MANAGEMENT_SCOPE=2
```
 - b. Run the CompleteMigration action on the same node to complete the migration of the peer domain. If migrating to a PTF, the PTF must be committed on all nodes before running the CompleteMigration action.

```
runact -c IBM.PeerDomain CompleteMigration Options=0
```

If the command is run before all the nodes are upgraded or the peer domain has less than half of its nodes online, an error message will result and the RSCTActiveVersion will remain unchanged. Upgrade all the nodes to a new level and make sure that half of the peer domain's nodes are online before executing the command again.

Creating a peer domain

To configure nodes into an RSCT peer domain, you need to:

- prepare initial security environment on each node that will be in the peer domain using the **preprnode** command.
- create a new peer domain definition by issuing the **mkrpdomain** command.
- bring the peer domain online using the **startdomain** command.

In a peer domain, processor architecture and operating system are heterogeneous. Starting with version 2.3.2.0 of RSCT, peer domain nodes can run either AIX or Linux. AIX nodes will support any processor architecture supported by the AIX operating system. The supported Linux distributions are detailed in “Supported Linux distributions for RSCT 2.4.1.0” on page 16. (Please note, however, that products designed to run in a peer domain may not support the same heterogeneous environment as RSCT. Please refer to the specific exploiter’s documentation for information on supported processor architecture and operating systems.)

Step 1: prepare initial security environment on each node that will participate in the peer domain

Before you can create your peer domain using the **mkrpdomain** command (described in “Creating a peer domain” on page 25), you first need to run the **preprnode** command to establish the initial trust between each node that will be in the peer domain, and the node from which you will run the **mkrpdomain** command. Later, when you run the **mkrpdomain** command, the configuration resource manager will establish the additional needed security across all peer domain nodes. This will enable you to issue subsequent commands from any node in the peer domain.

Note: The **preprnode** command will automatically exchange public keys between nodes. If you do not feel the security of your network is sufficient to prevent address and identity spoofing, you should refer to “Guarding against address and identify spoofing when transferring public keys” on page 157. If you are not sure if your network is secure enough, consult with a network security specialist to see if you are at risk.

The node from which you will issue the **mkrpdomain** command is called the *originator node*. Be aware that the originator node does not have to be a node you intend to include in your RSCT peer domain; it could be just a node where you issue the **mkrpdomain** command. It could, for example, be the management server of a management domain. To establish trust between the originator node and each node that will be in the peer domain, you must run the **preprnode** command on each node that will be in the peer domain. You will need to specify the name of the originator node as the parameter.

For example, say you will be issuing the **mkrpdomain** command on *nodeA*. From each node that will be in the peer domain, issue the command:

```
preprnode nodeA
```

You can also specify multiple node names on the command line:

```
preprnode nodeA nodeB
```

Instead of listing the node names on the command line, you can, using the **-f** flag, specify the name of a file that lists the node names. For example:

```
preprnode -f node.list
```

When using the **preprnode** command, you can identify the node by its IP address or by the long or short version of its DNS name. The **preprnode** command establishes the initial security environment needed by the **mkrpdomain** command by:

- retrieving the originator node's public key and adding it to the trusted host list of the local node. For more information about public keys and trusted host list files, refer to Chapter 6, "Understanding and administering cluster security services," on page 147.
- modifying the local node's RMC Access Control List (ACL) to enable access to its resources from the originator node. For more information about RMC ACL files, refer to "Managing user access to resources using RMC ACL files" on page 77.

You can specify multiple nodes on the **preprnode** command, in which case the initial trust will be established between the local node and each of the remote nodes listed. As long as you know which node will be the originator node, however, there should not be a need to specify multiple nodes on the **preprnode** command.

If you have, for security reasons, already manually transferred the public keys, you need to use the **-k** flag when you issue the **preprnode** command. For example:

```
preprnode -k nodeA nodeB
```

Using the **-k** flag disables the automatic transfer of public keys. You may also want to use the **-k** flag if you know the originator node and the local node have already been configured by CSM as part of the same management domain. In this case, the necessary public key transfer has already occurred. While allowing the **preprnode** command to copy the public key again will not result in an error, you could reduce overhead by disabling the transfer.

Although the **-k** flag disables automatic public key transfer, the **preprnode** command will still modify the node's RMC ACL file to enable access to the other nodes you will include in the peer domain.

For more information on security issues related to the automatic transfer of public keys, refer to Chapter 6, "Understanding and administering cluster security services," on page 147.

For complete syntax information on the **preprnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Once you have run the **preprnode** command on each node you will include in the peer domain, you can create a new peer domain using the **mkrpdomain** command (described next in "Step 2: create a new peer domain").

Step 2: create a new peer domain

The **mkrpdomain** command creates a new peer domain definition. A peer domain definition consists of:

- a peer domain name
- the list of nodes included in that peer domain
- the UDP port numbers to be used for Topology Services and Group Services daemon to daemon communication

For example, say you want to establish a peer domain with three nodes, and the nodes are identified by the DNS names *nodeA*, *nodeB*, and *nodeC*. Say also that,

when you issued the **preprnode** command from the nodes that will make up your peer domain, you determined that *nodeA* would be the originator node. To create a peer domain named *AppDomain*, you would, from *nodeA*, issue the command:

```
mkrpdomain AppDomain nodeA nodeB nodeC
```

The characters used for your domain name are limited to the ASCII characters A-Z, a-z, 0-9, . (period), and _ (underscore). The above command creates the peer domain definition *AppDomain* consisting of the nodes *nodeA*, *nodeB*, and *nodeC*.

Instead of listing the node names on the command line, you can use the **-f** flag to specify the name of a file that lists the node names. For example:

```
mkrpdomain -f node.list AppDomain
```

The configuration resource manager will at this time create the communication group definitions needed to later enable liveness checks (known as *heartbeating* in Topology Services) between the nodes of a peer domain. The configuration resource manager will attempt to automatically form a communication group based on subnets and inter-subnet accessibility. Each communication group is identified by a unique name. The name is assigned sequentially by suffixing CG with *existing highest suffix + 1*, such as CG1, CG2, and so on.

When you run the **startprdomain** command (described next in “Step 3: bring the peer domain online” on page 29), the configuration resource manager will supply the communication group definition information to Topology Services. For more information on Topology Services, refer to Chapter 7, “The Topology Services subsystem,” on page 167.

Since, in the preceding commands, a quorum type was not specified, a default quorum type will be used to calculate startup quorum, configuration quorum, and operational quorum. The default quorum type will depend on your environment. For most clusters, the default quorum type will be “Normal”. In OS/400 environments, the default will be “Override”. In environments with the IBM TotalStorage SAN File System, the default will be “SANFS”. For a description of the quorum types and how startup quorum, configuration quorum, and operational quorum are calculated for each type, refer to “What are quorum types?” on page 21.

To specify a quorum type, you can use the **-Q** flag followed by an integer or name indicating the quorum type. The quorum types are described in “What are quorum types?” on page 21. On the **mkrpdomain** command, you can specify the quorum type to be one of the following:

- 0 or “Normal”
- 1 or “Quick”

Note: The quorum types 3 (Override) and 4 (SANFS) are defined only for a few dedicated and embedded environments. You will not need to explicitly set the quorum type to either of these values.

For example, to specify quick startup mode, which is useful for large clusters, you could specify:

```
mkrpdomain -Q 1 AppDomain nodeA nodeB nodeC
```

or

```
mkrpdomain -Q Quick AppDomain nodeA nodeB nodeC
```

When starting a peer domain (as described next in “Step 3: bring the peer domain online”), you can override the quorum type to specify a different one for calculating startup quorum. You can also modify the quorum type as described in “Changing a peer domain’s quorum type” on page 39.

If the **mkrpdomain** command fails on any node, it will, by default, fail for all nodes. You can override this default behavior using the **-c** flag. You might want to use this flag, for example, when creating larger peer domain configurations. If you are creating a peer domain consisting of a large number of nodes, the chances that the **mkrpdomain** command would fail on any one is greater. In such a case, you probably would not want the operation to fail for all nodes based on a single node failing. You would therefore enter:

```
mkrpdomain -c -f node.list AppIDomain
```

Since, in the preceding commands, port numbers were not specified for Topology Services and Group Services daemon to daemon communication, the default port numbers (port 12347 for Topology Services and port 12348 for Group Services) will be used. You can override these defaults using the **mkrpdomain** command’s **-t** flag (to specify the Topology Services port) or **-g** flag (to specify the Group Services port). Any unused port in the range 1024 to 65535 can be assigned. For example:

```
mkrpdomain -t 1200 -g 2400 AppIDomain nodeA nodeB nodeC
```

For complete syntax information on the **mkrpdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Once you have created your peer domain definition using the **mkrpdomain** command, you can bring the peer domain online using the **startpdomain** command (described next in “Step 3: bring the peer domain online”).

Step 3: bring the peer domain online

The **startpdomain** command brings a peer domain online by starting the resources on each node belonging to the peer domain. To bring the peer domain online, simply pass the **startpdomain** command the name of a peer domain you have already defined using the **mkrpdomain** command. For example, to bring the peer domain *AppIDomain* online, you would, from any of the nodes in the peer domain, issue the command:

```
startpdomain AppIDomain
```

The peer domain’s quorum type (as described in “What are quorum types?” on page 21) will determine the startup quorum needed for bringing the peer domain online. The cluster’s quorum type will either be the default for your environment, or one you specified using the **mkrpdomain** command’s **-Q** flag (as described in “Step 2: create a new peer domain” on page 27). When starting a peer domain, you can also, if the quorum type is set to 0 (Normal) or 1 (Quick), override the quorum type to specify a different one for calculating startup quorum. Using the **startpdomain** command’s **-Q** flag, you can specify the startup quorum type to be either:

- 0 or “Normal”
- 1 or “Quick”

For example, if the quorum type is 0 (Normal), you could override that quorum type to specify that quick startup mode should be used to calculate startup quorum.

```
startpdomain -Q 1 AppIDomain
```

or

```
starttrpdomain -Q Quick App1Domain
```

Notes:

1. You cannot modify the startup quorum type if it has been implicitly set to 2 (Override) or 3 (SANFS).
2. You cannot specify the startup quorum type to be 2 (Override) or 3 (SANFS).

When bringing the peer domain online, the **starttrpdomain** command uses the peer domain configuration information you defined when you issued the **mkrpdomain** command. If necessary, the configuration resource manager will start Group Services and Topology Services on each of the nodes in the peer domain. The configuration resource manager will also at this time supply Topology Services with the communication group definition information for the peer domain. A communication group controls how liveness checks (*heartbeating* in Topology Services) are performed between the communications resources within the peer domains. The communication group also determines which devices are used for heartbeating in the peer domain. Each communication group has several characteristics. These characteristics specify:

- the number of missed heartbeats that constitute a failure
- the number of seconds between the heartbeats
- whether or not broadcast should be used
- whether or not source routing should be used

Each communication group also has a list of its member network interfaces.

To determine what communication groups were created, use the **lscomg** command (as described in “Listing communication groups” on page 40). The **lscomg** command not only lists the communication groups in your peer domain but also shows the characteristics about those communication groups. This means that even if the communication group was created automatically, you can use the **lscomg** command to see its default characteristics. If you would like to modify any of these characteristics, you can use the **chcomg** command as described in “Modifying a communication group’s characteristics” on page 41. To modify network interfaces in the communication group, refer to “Modifying a communication group’s network interface” on page 43.

By default, the **starttrpdomain** command will not attempt to bring the peer domain online until at least half the nodes have been contacted. The configuration resource manager searches for the most recent version of the peer domain configuration which it will use to bring the peer domain online. If you want the configuration resource manager to contact all nodes in the peer domain before bringing the domain online, specify the **starttrpdomain** command’s **-A** flag. This option is useful if you want to be sure that the most recent configuration is used to start the peer domain. For example:

```
starttrpdomain -A App1Domain
```

If you want the configuration resource manager to get the most recent configuration information from the local node only, specify the **starttrpdomain** command’s **-L** flag. For example:

```
starttrpdomain -L App1Domain
```

The configuration resource manager will not try to contact nodes to determine the latest configuration beyond a specified timeout value which is, by default, 120

seconds. If at least half the nodes (or all nodes if you have specified the **-A** flag) have not been contacted in that time, the configuration resource manager will not start the peer domain. You can, however, increase the timeout value using the **startprdomain** command's **-t** flag. For example, to have the operation time out at 240 seconds, you would issue the command:

```
startprdomain -t 240 ApplDomain
```

After the domain is brought online, you can use the **lsrpnnode** command to list information about the nodes in the domain. You can run this command from any node in the peer domain. Results are similar to the following.

Name	OpState	RSCTVersion
nodeA	online	2.2.1.20
nodeB	online	2.2.1.20
nodeC	online	2.2.1.20
nodeD	offline	2.2.1.20
nodeE	offline	2.2.1.20

You can also view all the network interfaces in the domain by issuing the **lsrsrc** command. Before issuing this generic RMC command, you should first set the management scope to 2 to indicate it is a peer domain, as follows:

```
export CT_MANAGEMENT_SCOPE=2
```

Then you can view the network interfaces in the peer domain by issuing the command:

```
lsrsrc -a IBM.NetworkInterface
```

Note: When you use the **-a** flag on the **lsrsrc** command, the **lsrsrc** command will automatically set the **CT_MANAGEMENT_SCOPE** environment variable. The only time you need to explicitly set the **CT_MANAGEMENT_SCOPE** environment variable is if the node is in both a peer domain and a management domain.

When a node becomes a member of the peer domain, it is assigned a unique integer which is referred to as a “node number”. Node numbers are used on certain commands and by some subsystems (for example, Topology Services). To view the node numbers, issue the following command from any online node in the peer domain. The attribute **NodeList** identifies the node numbers of all the nodes defined in the online cluster.

```
lsrsrc -a IBM.PeerNode Name NodeList
```

Note: We recommend that, once a peer domain is created and the peer nodes are online, you save a record of the node to node number mapping. Such a record may be helpful if you later need to restore nodes with their original node numbers (as described in the *Reliable Scalable Cluster Technology: Diagnosis Guide*). To save a record of the node to node number mapping, issue the following command from a node that is online in the peer domain.

```
lsrsrc -x -D' ' IBM.PeerNode Name NodeList | sed 's/{/ /g' | \
sed 's/}/ /g'|sed 's"/ //g' > rpdNodeMap.save
```

You can later take the peer domain offline using the **stopprdomain** command. You can also take an individual node offline using the **stopprnode** command. These commands are described in “Taking individual nodes of a peer domain, or an entire peer domain, offline” on page 36.

For complete syntax information on the **startprdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Adding nodes to an existing peer domain

“Creating a peer domain” on page 25 describes the initial setup of a peer domain. This section describes how to add new nodes to an existing peer domain. To add a node to a peer domain, you need to:

- prepare security on the node using the **preprnode** command
- add the node to the peer domain definition using the **addrnode** command
- bring the node online in the peer domain using the **startprnode** or **startprdomain** command

Step 1: prepare security environment on the node

Before you can add a node to a peer domain using the **addrnode** command (described next in “Step 2: add node to the peer domain” on page 34), you first need to issue the **preprnode** command to establish the initial trust between the node to be added, and the node from which you will issue the **addrnode** command. Later, when you issue the **addrnode** command, the configuration resource manager will establish the additional security environment so that the new node can issue subsequent configuration resource manager commands.

The node from which you will issue the **addrnode** command is called the *originator node*, and must be a node that is already part of the RSCT peer domain. To establish trust between the originator node and the node to be added to the peer domain, you must first run the **preprnode** command on the node to be added. On the **preprnode** command, you must either specify all the existing nodes in the peer domain, or else you must specify the Configuration Manager group leader. If the peer domain does not consist of many nodes, you will probably find it easiest to specify all the existing nodes on the **preprnode** command. For example, if the peer domain consists of *nodeA*, *nodeB*, and *nodeC*, you would enter the following on the node you wish to add to the peer domain:

```
preprnode nodeA nodeB nodeC
```

You identify the nodes by their IP addresses or by the long or short version of their DNS names.

If you are unsure which nodes are in a peer domain, enter the **lsrnode** command from a node that is active in the peer domain.

```
lsrnode
```

Output is similar to:

Name	OpState	RSCTVersion
nodeA	Online	2.3.3.0
nodeB	Online	2.3.3.0
nodeC	Online	2.3.3.0

Instead of listing the node names on the command line, you can, using the **-f** flag, specify the name of a file that lists the node names or IP addresses. When the peer domain consist of a large number of nodes, you may find listing the nodes in a file easier than entering them all on the command line. For example, if the nodes were listed in the file *node.list*, you would enter the following command on the node you will be adding to the peer domain:

```
preprnode -f node.list
```

An easy way to generate the *node.list* file used in the preceding example, would be to enter the following command on a node that is online in the peer domain:

```
lsrpnode -x | awk '{print $1}' > node.list
```

Once the file is generated, send it to the new node on which you will enter the **preprnode** command.

Another method that you may find easier when adding a node to a large peer domain, is to specify the peer domain's Group Leader on the **preprnode** command. Specifying the Group Leader eliminates the need to specify all the nodes in the peer domain. A Group Leader is a Topology Services and Group Services term for a coordinating node of Configuration Manager group. Although the operation of the Topology Services and Group Services subsystems should be transparent to you, they are used by a peer domain for distributed coordination and synchronization. For more information on Topology Services and Group Services, refer to Chapter 7, "The Topology Services subsystem," on page 167 and Chapter 8, "The Group Services subsystem," on page 185.

To find out which node in the peer domain is the Group Leader, enter the following SRC command on a node that is online in the peer domain:

```
lssrc -ls IBM.ConfigRM
```

Results will be similar to the following. Make note of the Group Leader (highlighted in bold text in this example).

```
Subsystem      : IBM.ConfigRM
PID            : 17880
Cluster Name   : Zagreus
Node Number    : 1
Daemon start time : Mon Oct 20 22:01:43 EDT 2003
```

Daemon State: Online in JoeD

```
ConfigVersion: 0x53fb2ff09
Group IBM.ConfigRM:
  Providers: 2
  GroupLeader: node8, 0x9a6befe2be807d07, 1
```

```
Information from malloc about memory use:
Total Space   : 0x009c0480 (10224768)
Allocated Space: 0x0086fad8 (8846040)
Unused Space  : 0x0014e3e0 (1369056)
Freeable Space : 0x00000000 (0)
```

Supply the name of the Group Leader node on the **preprnode** command. Specifying the Group Leader node eliminates the need to specify the other nodes in the peer domain.

```
preprnode node8
```

If you have chosen, for security reasons, to manually transfer the public keys, you need to use the **-k** flag when you issue the **preprnode** command. For example:

```
preprnode -k nodeA nodeB nodeC
```

Using the **-k** flag disables the automatic transfer of public keys. You may also want to use the **-k** flag if you know the originator node and local node have already been configured by CSM as part of the same management domain. In this case, the necessary public key transfer has already occurred. While allowing the **preprnode** command to copy the public key again will not result in an error, you could reduce overhead by disabling the transfer.

Although the **-k** flag disables the public key transfer, the **preprnode** command will still modify the node's RMC ACL file to enable access to the other nodes in the peer domain.

For information on security issues related to the automatic transferal of public keys, refer to "Guarding against address and identify spoofing when transferring public keys" on page 157.

For complete syntax information on the **preprnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Once you have set up the security environment on the node, you can add it to the peer domain using the **addrpnode** command.

Step 2: add node to the peer domain

When you initially set up an RSCT peer domain (described in "Creating a peer domain" on page 25), you use the **mkrpdomain** command to create the initial peer domain definition. To now add one or more nodes to that existing peer domain definition, you use the **addrpnode** command, passing it the IP address or DNS name of the node you wish to add. Keep in mind, however, that any change to the online cluster definition requires a *configuration quorum* of $(n/2)+1$ nodes (where n is the number of nodes defined in the cluster) to be active. In other words, you can not change an online cluster definition unless a majority of the nodes are online in the domain.

To add the node whose DNS name is *nodeD* to a peer domain, issue the following command from a node in the peer domain:

```
addrpnode nodeD
```

You can also add multiple nodes to the peer domain definition. You can do this either by listing them all on the command line:

```
addrpnode nodeD nodeE
```

Or else you can, using the **-f** flag, specify the name of a file that lists the node names:

```
addrpnode -f node.list
```

The configuration resource manager will at this time modify the communication group definitions needed later to extend liveness checks (Topology Services' "heartbeating") to the new nodes. When you issue the **startpnode** command (described next in "Step 3: bring node online in the peer domain" on page 35), the configuration resource manager will supply the modified communication group definition information to Topology Services. For more information on communication groups, refer to "Understanding and working with communication groups" on page 39. For more information on Topology Services, refer to Chapter 7, "The Topology Services subsystem," on page 167.

For complete syntax information on the **addrpnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Once you have added a node to an existing peer domain definition using the **addrpnode** command, you can bring the node online using the **startpnode** or

startprdomain command. These commands are described next in “Step 3: bring node online in the peer domain.”

Step 3: bring node online in the peer domain

The **startprnode** command brings an offline node online in the current peer domain. To see which nodes are currently defined in the peer domain, use the **lsrprnode** command from any node in the peer domain.

```
lsrprnode
```

Issuing this command lists information about the nodes defined in the peer domain. For example:

Name	OpState	RSCTVersion
nodeA	online	2.2.1.20
nodeB	online	2.2.1.20
nodeC	online	2.2.1.20
nodeD	offline	2.2.1.20
nodeE	offline	2.2.1.20

In this example, *nodeD* and *nodeE* are currently offline. Before you bring them online in the current RSCT peer domain, you might want to check that the nodes are not online in another RSCT peer domain. A node can be defined to more than one peer domain, but can be online in only one at a time. If you issue the **startprnode** command for a node that is already online in another peer domain, the node will not be brought online in the new peer domain, but will instead remain online in the other peer domain. To list peer domain information for a node, use the **lsrpdomain** command. For example, to determine if *nodeD* is currently online in any other peer domain, issue the following command on *nodeD*:

```
lsrpdomain
```

Issuing this command lists information about the peer domains a node is defined in. For example:

Name	OpState	RSCTActiveVersion	MixedVersions	TSPort	GSPort
App1Domain	offline	2.2.1.20	no	12347	12348

This output shows us that *nodeD* is not defined in any other peer domain, and so cannot be online in any other peer domain. To bring it online in the current peer domain, issue the command from any online node.

```
startprnode nodeD
```

The configuration resource manager will at this time supply Topology Services on the new node with the latest cluster definition for the peer domain. This will extend the Topology Services liveness checks to the new node.

If there are multiple nodes offline in the peer domain, you can also use the **startprdomain** command to bring all of the offline nodes online in this peer domain. For example, to bring the peer domain *App1Domain* online, you would, from any node, issue the command:

```
startprdomain App1Domain
```

All the offline nodes, if not already online in another peer domain, will be invited to go online.

For more information about the **startprdomain** command, refer to the directions for creating a peer domain (the **startprdomain** command is described in more detail in “Step 3: bring the peer domain online” on page 29 of those directions). For complete syntax information on the **startprnode**, **startprdomain**, **lsrprnode**, or

lsrpdomain commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Taking individual nodes of a peer domain, or an entire peer domain, offline

In order to perform node maintenance or make application upgrades, you might want to take individual nodes of a peer domain, or an entire peer domain, offline. This section describes how to:

- Take a peer domain node offline using the **stoprnode** command
- Take a peer domain offline using the **stoprpdomain** command

Taking a peer domain node offline

The **stoprnode** command takes one or more nodes of a peer domain offline. You might need to do this to perform application upgrades, to perform maintenance on a node, or prior to removing the node from the peer domain (as described in “Removing a node from a peer domain” on page 37). Also, since a node may be defined in multiple peer domains, but online in only one at a time, you might need to take a node offline in one peer domain so that you may bring it online in another. To take a node offline, issue the **stoprnode** command from any online node in the peer domain, and pass it the peer domain node name of the node to take offline.

You can list the peer domain node names by issuing the **lsrnode** command for any node in the peer domain:

```
lsrnode
```

Issuing this command lists information about the nodes defined in the peer domain. This information includes the peer domain node names. For example:

Name	OpState	RSCTVersion
nodeA	offline	2.2.1.20
nodeB	online	2.2.1.20
nodeC	online	2.2.1.20
nodeD	online	2.2.1.20
nodeE	offline	2.2.1.20

To take the node whose peer domain node name is *nodeA* offline, you would issue the following command from any online node:

```
stoprnode nodeA
```

You can also take multiple nodes offline. For example:

```
stoprnode nodeA nodeB
```

An RSCT subsystem (such as Topology Services or Group Services) may reject the **stoprnode** command’s request to take a node offline if a node resource is busy. To force the RSCT subsystems to take the node offline regardless of the state of node resources, use the **stoprnode** command’s **-f** flag. For example:

```
stoprnode -f nodeA
```

To later bring the node back online, use the **startprnode** command as described in “Step 3: bring node online in the peer domain” on page 35. For complete syntax information on the **stoprnode** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Taking a peer domain offline

In order to perform maintenance on a peer domain, you might wish to take it offline. To take a peer domain offline, issue the **stoprpdomain** command from any online node in the peer domain. You pass the **stoprpdomain** command the name of the peer domain you wish to take offline. For example, to take all the nodes in the peer domain *AppIDomain* offline:

```
stoprpdomain AppIDomain
```

An RSCT subsystem (such as Topology Services or Group Services) may reject the **stoprpdomain** command's request to take a peer domain offline if a peer domain resource is busy. To force the RSCT subsystems to take the peer domain offline regardless of the state of peer domain resources, use the **stoprpdomain** command's **-f** flag. For example:

```
stoprpdomain -f AppIDomain
```

Stopping a peer domain does not remove the peer domain definition; the peer domain can therefore be brought back online using the **startrpdomain** command. For more information on the **startrpdomain** command, refer to “Step 3: bring the peer domain online” on page 29. For complete syntax information on the **stoprpdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing individual nodes from, or removing an entire, peer domain

When upgrading hardware or otherwise reorganizing your peer domain configuration, you may need to remove individual nodes from a peer domain, or else remove an entire peer domain definition. This section describes how to:

- remove a node from a peer domain using the **rmpnode** command
- remove a peer domain definition using the **rmpdomain** command

Removing a node from a peer domain

In order to remove a node from a peer domain, the node must be offline. If the node you wish to remove is not currently offline, you must use the **stoprpdomain** command to take it offline. For more information on the **stoprpdomain** command, refer to “Taking a peer domain node offline” on page 36.

To see if the node is offline, issue the **lsrpdnode** command from any node in the peer domain.

```
lsrpdnode
```

Issuing this command lists information about the nodes defined in the peer domain. For example:

Name	OpState	RSCTVersion
nodeA	offline	2.2.1.20
nodeB	online	2.2.1.20
nodeC	online	2.2.1.20
nodeD	online	2.2.1.20
nodeE	offline	2.2.1.20

In this example, *nodeA* and *nodeE* are offline and can be removed. To remove a node, issue the **rmpnode** command from any online node in the peer domain, passing the **rmpnode** command the peer domain node name of the node to remove. For example, to remove *nodeA*:

```
rnrnode nodeA
```

You can also remove multiple nodes from the peer domain:

```
rnrnode nodeA nodeE
```

Since removing a node changes the domain configuration definition, the **rnrnode** command, by default, requires a configuration quorum. The configuration quorum for this command is either a majority of nodes or exactly half the nodes provided the configuration resource manager can remove the configuration from at least one of the offline nodes. You can override the need for a configuration quorum and force node removal by specifying the **-f** option on the **rnrnode** command. For example:

```
rnrnode -f nodeA
```

For complete syntax information on the **rnrnode** and **lsrnode** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing a peer domain

Removing a peer domain involves removing the peer domain definition from each node on the peer domain.

You can remove the peer domain definition by issuing the **rnrpdomain** command from any online node in the peer domain. You pass the **rnrpdomain** command the name of the peer domain. For example, to remove the peer domain *App1Domain*:

```
rnrpdomain App1Domain
```

The **rnrpdomain** command removes the peer domain definition on all of the nodes that are reachable from the node where the command was issued. If all the nodes are reachable, then the command will attempt to remove the peer domain definition from all nodes. If a node is not reachable from the node where the **rnrpdomain** is run (for example, the network is down or the node is inoperative), the **rnrpdomain** command will not be able to remove the peer domain definition on that node. If there are nodes that are not reachable from the node where the **rnrpdomain** command was run, you will need to run the **rnrpdomain** command from each node that did not have their peer domain definition removed. You should include the **-f** option to force the removal:

```
rnrpdomain -f App1Domain
```

You can also use the **-f** flag if an RSCT subsystem (such as Topology Services or Group Services) rejects the **rnrpdomain** command because a peer domain resource is busy. The **-f** flag will force the RSCT subsystems to take the peer domain offline and remove the peer domain definitions regardless of the state of peer domain resources.

For complete syntax information on the **rnrpdomain** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Changing a peer domain's quorum type

As described in “What are quorum types?” on page 21, a peer domain's quorum type is used to calculate startup quorum, configuration quorum, and operational quorum. The peer domain's quorum type will either be the default for your environment, or one you explicitly specify. When creating a peer domain, you can specify the quorum type using the **mkrpdomain** command's **-Q** flag.

Once a peer domain is created, you can also modify its quorum type using the generic RMC command **chrsrc**. You can use the **chrsrc** command to modify the QuorumType attribute of the PeerNode class.

For example, to modify a peer domain's quorum type to quick startup mode, you would enter the following command from a node that is online in the peer domain.

```
chrsrc -c IBM.PeerNode QuorumType=1
```

For detailed syntax information on the **chrsrc** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Understanding and working with communication groups

Communication groups control how liveness checks (in other words, Topology Services' “heartbeats”) are performed between the communication resources within the peer domain. Each communication group corresponds to a Topology Services heartbeat ring. It identifies the attributes that control the liveness checks between the set of network interfaces and other devices in the group.

The configuration resource manager automatically forms communication groups when a new peer domain is formed by the **mkrpdomain** command. When you bring a peer domain online using the **startrpdomain** command, the configuration resource manager will supply the communication group definition to Topology Services which will create the actual heartbeat rings needed to perform liveness checks for the peer domain nodes. The configuration resource manager may also form new communication groups as new nodes are added to the peer domain by the **addrpnode** command. When these added nodes are brought online by the **startrpnode** command, the configuration resource manager supplies the modified information to Topology Services which may modify existing heartbeat rings or create additional heartbeat rings.

The configuration resource manager's automatic creation of communication groups is based on subnet and intersubnet accessibility. For each communication group, the goal is to define a set of adapters (with no more than one adapter from each node), each having end-to-end connectivity with the others. Given the restriction that at most one adapter from each node can belong to a given communication group:

- all adapters in the same subnet will be in the same communication group, unless one node has multiple adapters in the same subnet.
- adapters in different subnets that can communicate with each other may be in the same communication group if they have connectivity.

The configuration resource manager allows you to create your own communication groups and also change the adapter membership in an existing communication group. However, since the configuration resource manager will create the

communication groups automatically, such manual configuration is neither necessary or advisable. **Manual configuration may be exercised, but only in unavoidable situations** (such as when a network configuration is more complex than our automatic communication group creation algorithm has anticipated and can handle). Manual configuration changes that do not conform to the above rules and restrictions may cause partitioning of the peer domain. For more information, refer to “Manually configuring communication groups” on page 43.

When the configuration resource manager automatically creates communication groups, it gives them default characteristics such as:

- Sensitivity — the number of missed heartbeats that constitute a failure.
- Period — the number of seconds between the heartbeats.
- Priority — the importance of this communication group with respect to others.
- Broadcast/No Broadcast — whether or not to broadcast (if the underlying network supports it).
- Enable/Disable Source Routing — In case of adapter failure, whether or not source routing should be used (if the underlying network supports it).

You can modify a communication group’s characteristics using the **chcomg** command as described in “Modifying a communication group’s characteristics” on page 41.

Listing communication groups

The **lscmg** command lists information about the communication groups in a peer domain. It lists the:

- name of the communication group
- the sensitivity setting (the number of missed heartbeats that constitute a failure)
- the period setting (the number of seconds between heartbeats)
- the priority setting (the relative priority of the communication group)
- whether or not broadcast should be used if it is supported by the underlying media
- whether or not source routing should be used if it is supported by the underlying media
- the path to the Network Interface Module (NIM) that supports the adapter types in the communication group
- the NIM start parameters
- the name of the resource interface that refers to this communication group
- the peer domain node name of the resource interface that refers to this communication group
- the IP address of the resource interface that refers to this communication group
- the subnet mask of the resource interface that refers to this communication group
- the subnet of the resource interface that refers to this communication group

For example, to list general information about the peer domain *ApplDomain*, enter the following command from a node that is online to *ApplDomain*:

```
lscmg
```

The configuration resource manager lists information about the communication groups defined in the peer domain:

```

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

```

If there are multiple communication groups defined on the node, and you want only a particular one listed, specify the name of the communication group on the **lscmg** command. For example, to list information about the communication group *ComGrp*, enter:

```
lscmg ComGrp
```

To list interface resource information for a communication group, use the **-i** flag on the **lscmg** command.

```
lscmg -i ComGrp1
```

Output is similar to:

```

IName  IHostName  IIPAddr  ISubnetMask  ISubnet
eth0   n24.ibm.com  9.234.32.45  255.255.255.2  9.235.345.34
eth0   n25.ibm.com  9.234.32.46  255.255.255.2  9.235.345.34

```

If you want to change any of the settings of a communication group, you can use the **chcmg** command as described in “Modifying a communication group’s characteristics.” For complete syntax information on the **lscmg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Modifying a communication group’s characteristics

A communication group has a number of properties that determine its behavior. These properties are established when the communication group is created and include such tunables as the group’s sensitivity, period, and priority settings. Using the **chcmg** command, you can change the settings, and so the behavior, of a communication group. To see the current settings for a communication group, use the **lscmg** command as described in “Listing communication groups” on page 40.

You can also use the **chcmg** command to modify a communication group’s network interface assignment. You typically do not need to modify this, and in fact should perform such manual configuration only in unavoidable situations. See “Modifying a communication group’s network interface” on page 43 for more information.

Since the **chcmg** command modifies the domain configuration definition, it will not change a communication group’s characteristics unless a majority of nodes are online in the domain. If such a *configuration quorum* exists, the domain configuration definition can be modified.

For complete syntax information on the **chcmg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Modifying a communication group’s sensitivity setting

A communication group’s sensitivity setting refers to the number of missed Topology Services’ heartbeats that constitute a failure. To determine what a communication group’s sensitivity setting is, use the **lscmg** command as described in “Listing communication groups” on page 40. To modify a communication group’s sensitivity setting, use the **chcmg** command with its **-s** flag. For example, to modify the

communication group *ComGrp1* so that its sensitivity setting is 4, issue the following command on a node that is online in the peer domain.

```
chcomg -s 4 ComGrp1
```

The sensitivity setting must be an integer greater than or equal to 2.

Modifying a communication group's period setting

A communication group's period setting refers to the number of seconds between Topology Service's heartbeats. To determine what a communication group's period setting is, use the **lscmg** command as described in "Listing communication groups" on page 40. To modify a communication group's period setting, use the **chcomg** command with its **-p** flag. For example, to modify the communication group *ComGrp1* so that its period is 3, issue the following command on a node that is online in the peer domain.

```
chcomg -p 3 ComGrp1
```

The period setting must be an integer greater than or equal to 1.

Modifying a communication group's priority setting

A communication group's priority setting refers to the importance of this communication group with respect to others and is used to order the topology services heartbeat rings. The lower the number means the higher the priority. The highest priority is 1. To determine what a communication group's priority setting is, use the **lscmg** command as described in "Listing communication groups" on page 40. To modify a communication group's priority setting, use the **chcomg** command with its **-t** flag. For example, to modify the communication group *ComGrp1* so that its priority is 3, issue the following command on a node that is online in the peer domain.

```
chcomg -t 3 ComGrp1
```

Modifying a communication group's broadcast setting

A communication group's broadcast setting specifies whether or not broadcast will be used (provided the underlying network supports it). To determine what a communication group's broadcast setting is, use the **lscmg** command as described in "Listing communication groups" on page 40. To modify a communication group's broadcast setting so that broadcast operations are enabled, use the **chcomg** command with its **-b** flag. For example, to modify the communication group *ComGrp1* so that broadcast will be used (provided the underlying network supports it), issue the following command on a node that is online in the peer domain.

```
chcomg -b ComGrp1
```

To modify a communication group's broadcast setting so that broadcast operations are disabled, use the **chcomg** command with its **-x b** flag. For example, to modify the communication group *ComGrp1* so that broadcast will **not** be used, issue the following command on a node that is online in the peer domain.

```
chcomg -x b ComGrp1
```

Modifying a communication group's source routing setting

A communication group's source routing setting specifies whether or not source routing will be used in case of adapter failure (provided the underlying network supports it). To determine what a communication group's source routing setting is, use the **lscmg** command as described in "Listing communication groups" on page 40.

By default, source routing is enabled. To modify a communication group's broadcast setting so that source routing is disabled, use the **chcomg** command with its **-x r** flag. For example, to modify the communication group **ComGrp1** so that source routing will not be used, issue the following command on a node that is online in the peer domain.

```
chcomg -x r ComGrp1
```

To modify a communication group's source routing setting so that source routing is enabled, use the **chcomg** command with its **-r** flag. For example, to modify the communication group *ComGrp1* so that source routing will be used in case of adapter failure, issue the following command on a node that is online in the peer domain.

```
chcomg -r ComGrp1
```

Manually configuring communication groups

This section describes how to change the adapter membership of an existing communication group, create a new communication group, and remove communication groups. We would like to stress that such **manual configuration is, under normal circumstances, unnecessary and inadvisable**. Under normal circumstances, communication groups are automatically created when a new peer domain is formed by the **mkrpdomain** command, and modified when a node is added by the **addrpnode** command. When the peer domain is brought online by the **startpdomain** command or the new node is brought online by the **startpnode** command, the configuration resource manager supplies the communication group information to Topology Services which will create/modify the heartbeat rings.

Manual configuration may be exercised, but only in unavoidable situations (such as when a network configuration is more complex than our automatic communication algorithm has anticipated or can handle).

Note: The three configuration commands described in this section — **chcomg**, **mkcomg**, and **rmcomg** — all modify a domain's configuration definition and, for that reason, will not make any changes unless a majority of nodes are online in the domain. If such a *configuration quorum* exists, the domain configuration definition can be modified.

Modifying a communication group's network interface

"Modifying a communication group's characteristics" on page 41 describes how to use the **chcomg** command to modify a communication group's tunables (such as its sensitivity, period, and priority settings). You can also use the **chcomg** command to modify a communication group's network interface assignment. We do not recommend you do this, and any changes you make must conform to the following rules. These are the same rules that the configuration resource manager uses in creating communication groups automatically. Failure to follow these rules may cause partitioning of the peer domain. The rules are:

1. at most one adapter from each node can belong to a given communication group.
2. given the restriction in (1), all adapters in the same subnet will be in the same communication group.
3. given the restriction in (1), adapters on different subnets that can communicate with each other may be in the same communication group.

In addition, because RSCT uses IP broadcast to optimize its communication, the following rules should be followed when configuring network interfaces.

- For each network interface, its broadcast address or subnet mask should be consistent with each other. That is: **Bcast address = IP address OR (negated netmask)**. For example, if IP address = 1.2.3.4 and netmask = 255.255.255.0, then the broadcast address should be 1.2.3.255.
- The subnet mask and broadcast addresses should be the same across all the interfaces that belong to the same subnet. Interfaces that belong to different subnets are allowed to have different subnet masks.

To modify a communication group's network interface:

- assign the communication group to a network interface using either the **-i** flag or the **-S** flag with the **n** clause.
 - using the **-i** flag and **n** clause, you can assign the communication group to the network interface by specifying the network interface name and, optionally, the name of the node where the resource can be found.
 - using the **-S** flag with the **n** clause, you can assign the communication group to the network interface by specifying a selection string.
- If necessary, use the **-e** flag to specify the path to the Network Interface Module (NIM) that supports the adapter type, and the **-m** flag to specify any character strings you want passed to the NIM as start parameters. It is likely that the NIM path (which is `/usr/sbin/rsct/bin/hats_nim`) is already specified in the communication group definition; issue the **lscomg** command as described in "Listing communication groups" on page 40 to ascertain this.

For example, to modify the *ComGrp1* communication group's network interface to the network interface resource named *eth0* on *nodeB*, you would enter the following from a node that is online in the peer domain.

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

To specify the NIM path and options (in this case, the option is `"-l 5"` to set the logging level), you would enter the following from a node that is online in the peer domain.

```
chcomg -i n:eth0:nodeB -e /usr/sbin/rsct/bin/hats_nim -m "-l 5" ComGrp1
```

To assign the communication group *ComGrp1* to the network interface resource that uses the subnet 9.123.45.678, you would enter the following from a node that is online in the peer domain.

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

Creating a communication group

Under normal circumstances, the configuration resource manager creates communication groups automatically when a new peer domain is formed, and modifies them as new nodes are added to the peer domain. You should not need to create your own communication groups; this ability is provided only to address special situations such as when a network configuration is more complex than our automatic communication group algorithm has anticipated or can handle.

To create a communication group, use the **mkcomg** command. One of the key things you'll need to specify is the communication group's network interface assignment. When making such assignments, you must conform to the following rules. These are the same rules that the configuration resource manager uses when creating communication groups automatically. Failure to follow these rules may cause partitioning of the peer domain. The rules are:

1. at most one adapter from each node can belong to a given communication group.

2. given the restriction in (1), all adapters in the same subnet will be in the same communication group.
3. given the restriction in (1), adapters on different subnets that can communicate with each other may be in the same communication group.

To set a communication group's network interface:

- assign the communication group to a network interface using either the **-i** flag or the **-S** flag with the **n** clause.
 - using the **-i** flag and **n** clause, you can assign the communication group to the network interface by specifying the network interface name and, optionally, the name of the node where the resource can be found.
 - using the **-S** flag with the **n** clause, you can assign the communication group to the network interface by specifying a selection string.
- Use the **-e** flag to specify the path to the Network Interface Module (NIM). In RSCT, a NIM is a process started by the Topology Services' daemon to monitor a local adapter. The NIM executable is located at `/usr/sbin/rsct/bin/hats_nim`, and one instance of the NIM process exists for each local adapter that is part of the peer domain. In addition to the **-e** flag, you can use the **-m** flag to specify any character strings you want passed to the NIM as start parameters

For example, to create the communication group *ComGrp1*, specifying the network interface resource name *eth0* on *nodeB*, you would enter the following from a node that is online in the peer domain.

```
mkcomg -i n:eth0:nodeB -e /usr/sbin/rsct/bin/hats_nim -m "-l 5" ComGrp1
```

The NIM parameters in the preceding example (`-l 5`) set the logging level.

To create the communication group *ComGrp1*, specifying the network interface resource that uses the subnet 9.123.45.678, you would enter the following from a node that is online in the peer domain.

```
mkcomg -S n:"Subnet == 9.123.45.678" -e /usr/sbin/rsct/bin/hats_nim
-m "-l 5" ComGrp1
```

You can also set a number of tunables for the Topology Services' heartbeat ring when issuing the **mkcomg** command. You can specify the:

- sensitivity setting (the number of missed heartbeats that constitute a failure) using the **-S** flag.
- period setting (the number of seconds between the heartbeats) using the **-p** flag.
- priority setting (the importance of this communication group with respect to others) using the **-t** flag.
- broadcast setting (whether or not to broadcast if the underlying network supports it) using the **-b** (broadcast) or **-x b** (do not broadcast) flags.
- source routing setting (in case of adapter failure, whether or not source routing should be used if the underlying network supports it) using the **-r** (use source routing) or **-x r** (do not use source routing) flags.

For example, the following command creates the *ComGrp1* communication group as before, but also specifies that:

- its sensitivity is 4
- its period is 3
- its priority is 2
- broadcast should be used

- source routing should not be used

```
mkcomg -s 4 -p 3 -t 2 -b -x r -i n:eth0:nodeB -e /usr/sbin/rsct/bin/hats_nim
-m "-l 5" ComGrp1
```

You can display all of the settings for a communication group using the **lscmg** command (as described in “Listing communication groups” on page 40). To change any of the settings, you can use the **chcomg** command (as described in “Modifying a communication group’s characteristics” on page 41). For complete syntax information on the **mkcomg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing a communication group

The **rmcomg** command enables you to remove an already-defined communication group definition from a peer domain. As with all the manual configuration commands for communication groups, you will not normally need to do this. Manual configuration must be exercised with caution and only in unavoidable situations.

To list the communication groups in the peer domain, you can use the **lscmg** command as described in “Listing communication groups” on page 40. Before removing a communication group, you must first use the **chcomg** command to remove interface resource references to the communication group (as described in “Modifying a communication group’s network interface” on page 43).

To remove a communication group, simply supply its name to the **rmcomg** command. For example, to remove the communication group *ComGrp1*, issue the following command from a node that is online in the peer domain:

```
rmcomg ComGrp1
```

For complete syntax information on the **rmcomg** command, refer to its man page in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Modifying Topology Services and Group Services parameters

You can use the **chrsrc** command to change the control parameters used by Topology Services or Group Services for an online cluster through IBM.RSCTParameters resource class. For a complete discussion of Topology Services, refer to Chapter 7, “The Topology Services subsystem,” on page 167. For a complete discussion of Group Services, refer to Chapter 8, “The Group Services subsystem,” on page 185. To obtain more information on the IBM.RSCTParameters resource class, use the **lsrsrdef** command (as described in “Displaying attribute definition information for a resource or a resource class” on page 83).

An IBM.RSCTParameters resource class instance is created for each cluster when the cluster is first brought online. The control parameters include:

- Topology Services log size (TSLogSize)
- fixed priority (TSFixedPriority)
- pinned regions (TSPinnedRegions)
- Group Services log size (GSLogSize)
- maximum directory size (GSMaxDirSize)

An instance of the class is created automatically for a cluster when the cluster is brought online the first time. The default values for these parameters will be used when it is created.

To view or change the RSCT parameters, you use generic RMC commands (**lsrsrc** and **chrsrc** as described below). To use these generic RMC commands, you need to first set the management scope to 2.

```
export CT_MANAGEMENT_SCOPE=2
```

This tells RMC that the management scope is a peer domain.

To view the parameter values, issue the command:

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

These values are tunable. They can be changed using one of the following commands:

```
chrsrc -c IBM.RSCTParameters Attr=Value...
```

For example, to tell Topology Services to ping both code and data regions (a value of 3), execute the following command:

```
chrsrc -c IBM.RSCTParameters TSPinnedRegions=3
```

The command is equivalent to the Topology Services tunable command (**cthatstune**) or the Group Services tunable command (**cthagstune**).

Changing IP addresses in a peer domain

The configuration resource manager automatically monitors for configuration changes (such as IP address changes) in the RSCT peer domain. When such changes are detected, the configuration resource manager updates the online peer domain configuration to keep the configuration synchronized across all nodes of the peer domain. Since IP addresses are the critical path to a node, there are a couple of rules to follow when updating IP addresses so that the nodes in a peer domain can continue to be accessed by the configuration resource manager and other cluster subsystems. These rules are outlined in the following table.

Table 4. Changing IP Addresses in a Peer Domain

If a node has:	Then:
multiple IP addresses and you want to change only a subset of the IP addresses	There are no restrictions to changing IP addresses.

Table 4. Changing IP Addresses in a Peer Domain (continued)

If a node has:	Then:
multiple IP addresses and you want to change all the IP addresses on the node	<p>You must not change all the IP addresses at the same time. Leave at least one IP address unchanged so that communication to the node will not be lost. If communication to a node is lost, the other nodes in the domain will consider the changed node to be offline since they only know it by its old IP address. In addition, the configuration resource manager on the changed node will have no way of telling the remaining nodes about the change. To change IP addresses, you can either do so by changing the IP addresses one at a time, or change all but one in a single request. Once the node has been harvested after the first change and the cluster configuration is updated with the change, you can then proceed to modify the next or the last unchanged IP address. The configuration resource manager checks for changes periodically (every minute or so) and applies any detected changes to the cluster configuration. After making a change, you should wait about 1 minute and 30 seconds for the change to be reflected or until the command <code>lsrsrc IBM.NetworkInterface</code> reflects the change. Alternatively, you can force the configuration resource manager to detect the change by running the following command on the node where the IP address was changed.</p> <pre>refsrc IBM.NetworkInterface</pre>
single IP address	<p>This is the only access to the node. You should:</p> <ol style="list-style-type: none"> 1. Remove the node from the peer domain (using the rmrpnod command as described in “Removing a node from a peer domain” on page 37). 2. Change its IP address. 3. Add the node back to the peer domain. (Using the addrpnod command as described in “Adding nodes to an existing peer domain” on page 32).

Determining how your system responds to domain partitioning and subsystem daemon failure

In order to protect data, the configuration manager uses a quorum of nodes (called an *operational quorum*) to determine whether resources can be safely activated without creating conflicts with other subsystems. For more information, refer to “What is operational quorum?” on page 20.

This section describes the various ways you can configure your peer domain to determine how the configuration resource manager calculates operational quorum and responds to domain partitioning and subsystem daemon failure. The configuration tasks described in this section are all performed by issuing standard Resource Management and Control (RMC) commands such as **lsrsrc** and **chrsrc** to set attributes of various resources of the configuration resource manager. For this reason, it is important that you first understand RMC and how, along with the various resource managers, it enables you to manage the resources of your system in a consistent and generic manner. Refer to Chapter 4, “Managing and monitoring resources using RMC and resource managers,” on page 61 for more information.

This section describes how you can:

- determine the way critical resources are protected should a domain lose operation quorum or if the configuration manager, group services, or topology services daemons die or hang. This is done by setting the CritRsrcProtMethod attribute of the IBM.PeerNode class (or an individual IBM.PeerNode instance) and is described in “Setting the critical resource protection method for a peer domain or a node in a peer domain.”
- specify that the peer domain should always have operational quorum. Forcing operational quorum in this way, as opposed to having the configuration resource manager calculate whether the peer domain has operation quorum, is not recommended since it means that critical resource will not be protected. For more information, refer to “Overriding the configuration resource manager’s operational quorum calculation to force operational quorum” on page 51.
- set the active tie breaker that the configuration resource manager will use to resolve tie situations when two or more sub-domains containing exactly half the defined nodes are competing for operational quorum. In addition to describing how to set the active tie breaker, “Determining how the configuration resource manager will resolve tie situations when calculating operational quorum” on page 52 also describes how you can modify a tie-breaker definition, define a new tie breaker, explicitly resolve a tie when the active tie-breaker type is “Operator”.

For complete syntax information on the generic RMC commands (such as **lsrsrc** and **chrsrc**) described in this section, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Setting the critical resource protection method for a peer domain or a node in a peer domain

When an RSCT peer domain is partitioned into two or more sub-domains, the configuration resource manager will determine which sub-domain has operational quorum and will survive, and which others should be dissolved. If the sub-domain is to be dissolved, the configuration resource manager sets the OpQuorumState dynamic attribute of the PeerDomain resource to 2 (NoQuorum).

If critical resources are active on a node that has lost quorum (as indicated by the PeerNode resource’s CritRsrcActive dynamic attribute), the configuration resource manager uses a *critical resource protection method* on the node to ensure that critical resources are not corrupted as a result of the domain partitioning. This is essential, since certain applications require shared resource access. When a domain is partitioned, each sub-domain is unaware of any other sub-domain, and so multiple sub-domains may simultaneously access the shared resource and, in doing so, cause data corruption. A node’s critical resource protection method is also needed if the configuration manager, group services, or topology services daemons die or hang.

You can set the critical resource protection method for a peer domain by setting the CritRsrcProtMethod persistent attribute of the IBM.PeerNode resource class. By default, the same critical resource protection method will be employed for all nodes of the peer domain (all instances of the IBM.PeerNode resource class). You can specify a different critical resource protection method for a particular node, however, by setting the CritRsrcProtMethod persistent attribute for just that instance of the IBM.PeerNode resource class.

The following table shows the possible settings for the CritRsrcProtMethod persistent attribute.

Table 5. CritRsrcProtMethod Settings

CritRsrcProtMethod persistent attribute value:	Description
1	Hard reset and reboot.
2	Halt system.
3	Sync, hard reset and reboot.
4	Sync, Halt system.
5	None.
6	Exit and restart RSCT subsystems.

For the IBM.PeerNode resource class, the default value of CritRsrcProtMethod is 1 (hard reset and reboot). For the individual resource instances of IBM.PeerNode, the CritRsrcProtMethod persistent attribute can also have the value 0 which is the default and means that the resource instance inherits the value from the resource class.

To view or set the critical resource protection method for a peer domain or a node in the peer domain, use the standard RMC management commands **lsrsrc** and **chrsrc**.

For example, to list the current value of the CritRsrcProtMethod persistent attribute for each node in the domain, you would use the **lsrsrc** command.

```
# lsrsrc -t IBM.PeerNode Name CritRsrcProtMethod
Name          CritRsrcProtMethod
"Davros"      0
"Rassilon"    0
"Morbius"     0
"Zagreus"     0
```

The preceding output shows that each node currently inherits the peer domain's overall critical resource protection method. To list the domain-wide attributes, you would use the **lsrsrc** command with its **-c** flag.

```
# lsrsrc -c IBM.PeerNode
Resource Class Persistent Attributes for: IBM.PeerNode
resource 1:
  CommittedRSCTVersion = ""
  ActiveVersionChanging = 0
  OpQuorumOverride     = 0
  CritRsrcProtMethod   = 1
  OpQuorumTieBreaker   = "Fail"
```

To override the default domain-wide critical resource protection method on a single node, you would use the **chrsrc** command. This next example uses the **-s** flag and a selecting string to identify the node.

```
chrsrc -s"Name='Zagreus'" IBM.PeerNode CritRsrcProtMethod=3
```

To change the domain-wide critical resource protection method, you would use the **chrsrc** command with its **-c** flag.

```
chrsrc -c IBM.PeerNode CritRsrcProtMethod=3
```

For complete syntax information on the **lsrsrc** and **chrsrc** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or in the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Overriding the configuration resource manager's operational quorum calculation to force operational quorum

When a peer domain is partitioned, the configuration manager will, by default, determine which sub-domain has operational quorum using the following calculation:

```
If (( 2*numNodesOnline ) > numNodesDefined )
    OpQuorumState = HasQuorum
If (( 2*numNodesOnline ) == numNodesDefined )
    OpQuorumState = PendingQuorum
    (until tie breaker is won or lost).
If (( 2*numNodesOnline) < numNodesDefined )
    OpQuorumState = NoQuorum
```

By setting the `OpQuorumOverride` persistent class attribute of the `IBM.PeerNode` resource class, however, you can override this calculation and instead specify that the domain should always have operational quorum. If you do this, the `PeerDomain` resource's `OpQuorumState` dynamic attribute will always have the value 0 (HasQuorum). You should exercise caution before overriding the configuration resource manager's operational quorum calculation, since it means that critical resources will not be protected by the critical resource protection method.

The following table shows the possible settings for the `OpQuorumOverride` persistent class attribute of the `IBM.PeerNode` resource class.

Table 6. *OpQuorumOverride Settings*

OpQuorumOverride persistent class attribute value:	Description
0	Determine operation quorum.
1	Force operational quorum.

To view or set the `OpQuorumOverride` persistent class attribute of the `IBM.PeerNode` resource class, use the standard RMC management commands **lsrsrc** and **chsrc**.

For example, to list the current value of the `CritRsrcProtMethod` persistent attribute, you would use the **lsrsrc** command with its **-c** flag:

```
# lsrsrc -c IBM.PeerNode
Resource Class Persistent Attributes for: IBM.PeerNode
resource 1:
    CommittedRSCTVersion = ""
    ActiveVersionChanging = 0
    OpQuorumOverride = 0
    CritRsrcProtMethod = 1
    OpQuorumTieBreaker = "Fail"
```

To force operational quorum for the peer domain, you would use the **chsrc** command with its **-c** flag.

```
chsrc -c IBM.PeerNode OpQuorumOverride=1
```

For complete syntax information on the **lsrsrc** and **chsrc** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Determining how the configuration resource manager will resolve tie situations when calculating operational quorum

When a peer domain is partitioned, the configuration resource manager must determine which sub-domain has operational quorum and so will survive, and which sub-domain will be dissolved. Often, this is simply a case of determining which of the sub-domains has more than half of the nodes. In the case of a tie in which the peer domain has been partitioned into two sub-domains containing exactly half of the defined nodes, the configuration resource manager uses a tie-breaker resource (an instance of the IBM.TieBreaker resource class) to determine which sub-domain has operational quorum. A "tie" situation also occurs when exactly half the nodes of a domain are online, and the other half are inaccessible. You can have a number of IBM.TieBreaker resources defined, but only one can be active at any one time. This section describes how you can:

- set the active tie breaker for the peer domain.
- modify a tie-breaker resource definition
- define a new tie-breaker resource
- Explicitly resolve a tie

Setting the active tie breaker

The OpQuorumTieBreaker persistent class attribute of the IBM.PeerNode class indicates the active tie breaker for the peer domain. There may be a number of tie breakers (IBM.TieBreaker resources) defined for the peer domain, but only one may be active at a time. The configuration resource manager will use this active tie breaker after domain partitioning if there are multiple sub-domains with the same number of nodes to determine which sub-domain will have operational quorum. There are two predefined tie-breaker resources, and you can also define your own as described in "Defining a new tie breaker" on page 54. The two predefined tie breakers are described in the following table:

Table 7. Predefined Tie-Breakers (IBM.TieBreaker resources)

Tie Breaker	Description:
Operator	The system administrator resolves the tie by invoking the ResolveOpQuorumTie action of the IBM.PeerDomain resource class. Until the administrator explicitly breaks the tie, neither domain will have operational quorum. The OpQuorumState dynamic attribute of the PeerDomain resource will be 1 (PendingQuorum) until the administrator invokes the ResolveOpQuorumTie action. For more information, refer to "Explicitly resolving a tie when the active tie-breaker type is "Operator"" on page 59.
Fail	A pseudo tie breaker in that it does not actually resolve the tie situation. Neither sub-domain will have operational quorum. The OpQuorumState dynamic attribute of each PeerDomain resource will be 2 (NoQuorum). If critical resources are active on a domain that has lost quorum (as indicated by the PeerDomain resource's CritRsrcActive dynamic attribute), the configuration resource manager uses a critical resource protection method on the node to ensure that critical resources are not corrupted as a result of the domain partitioning. See "Setting the critical resource protection method for a peer domain or a node in a peer domain" on page 49 for more information on critical resource protection methods.

To view or set the active tie breaker (OpQuorumTieBreaker persistent class attribute of the IBM.PeerNode class), use the standard RMC management commands **lsrsrc** and **chsrc**.

For example, to list the current active tie breaker, you would use the **lsrsrc** command with its **-c** flag.

```
# lsrsrc -c IBM.PeerNode OpQuorumTieBreaker
Resource Class Persistent and Dynamic Attributes for: IBM.PeerNode
resource 1:
    OpQuorumTieBreaker = "Fail"
```

The preceding output shows us that the current active tie breaker is "Fail". To list the names of all of the available tie breaker resources, you would specify "Name" as a parameter on the **lsrsrc** command.

```
# lsrsrc IBM.TieBreaker Name
Resource Persistent and Dynamic Attributes for: IBM.TieBreaker
resource 1:
    Name = "Operator"
resource 2:
    Name = "Fail"
```

To make the "Operator" tie breaker the active tie breaker, you would use the **chrsrc** command with its **-c** flag.

```
chrsrc -c IBM.PeerNode OpQuorumTieBreaker="Operator"
```

If you set the active tie breaker to "Operator", then, should a tie situation occur, you will need to manually resolve the tie by invoking the `ResolveOpQuorumTie` action of the `IBM.PeerDomain` resource class. Refer to "Explicitly resolving a tie when the active tie-breaker type is "Operator"" on page 59 for more information.

For complete syntax information on the **lsrsrc** and **chrsrc** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Modifying a tie-breaker definition

A tie breaker (`IBM.TieBreaker` resource) has a number of persistent resource attributes that you can set to configure the tie breaker's behavior. To view or set these persistent class attributes, use the standard RMC management commands **lsrsrc** and **chrsrc**.

For example, to list the current persistent attribute values for all defined tie breakers, you would use the **lsrsrc** command.

```
# lsrsrc IBM.TieBreaker
Resource Persistent Attributes for: IBM.TieBreaker
resource 1:
    Name           = "Operator"
    Type           = "Operator"
    DeviceInfo     = ""
    ReprobeData    = ""
    ReleaseRetryPeriod = 0
    HeartbeatPeriod = 0
    PreReserveWaitTime = 0
    PostReserveWaitTime = 0
    NodeInfo       = {}
resource 2:
    Name           = "Fail"
    Type           = "Fail"
    DeviceInfo     = ""
    ReprobeData    = ""
    ReleaseRetryPeriod = 0
```

```

HeartbeatPeriod      = 0
PreReserveWaitTime   = 0
PostReserveWaitTime  = 0
NodeInfo              = {}

```

To limit the output of the **lsrsrc** command to display the persistent attribute values for only a particular tie breaker resource, you could use the **-s** flag and a selection string that identifies the particular tie breaker resource.

```

# lsrsrc -s"Name=='Operator'" IBM.TieBreaker
Resource Persistent Attributes for: IBM.TieBreaker
resource 1:
  Name              = "Operator"
  Type              = "Operator"
  DeviceInfo        = ""
  ReprobeData       = ""
  ReleaseRetryPeriod = 0
  HeartbeatPeriod   = 0
  PreReserveWaitTime = 0
  PostReserveWaitTime = 0
  NodeInfo          = {}

```

To obtain more information on any of these persistent attributes, use the **lsrsrcdef** command (as described in “Displaying attribute definition information for a resource or a resource class” on page 83). To change the persistent attributes of a tie breaker, the tie breaker must not be the active tie breaker. The OpQuorumTieBreaker persistent class attribute of the IBM.PeerNode class identifies the active tie breaker. If you are not sure if the tie breaker you want to modify is the active tie breaker, check the value of the OpQuorumTieBreaker persistent class attribute.

```

# lsrsrc -c IBM.PeerNode OpQuorumTieBreaker
Resource Class Persistent and Dynamic Attributes for: IBM.PeerNode
resource 1:
  OpQuorumTieBreaker = "Fail"

```

For instructions on the setting the OpQuorumTieBreaker persistent class attribute, refer to “Setting the active tie breaker” on page 52. As long as the tie breaker is not the active tie breaker, you can modify its persistent resource attributes using the **chrsrc** command. To identify a particular tie breaker, you will need to use the **chrsrc** command’s **-s** flag followed by a selection string that identifies the tie breaker resource. For example:

```

chrsrc -s"Name=='Operator'" IBM.TieBreaker ReleaseRetryPeriod=30

```

For complete syntax information on the **lsrsrc** and **chrsrc** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Defining a new tie breaker

In addition to the predefined tie breakers, you can also create your own by defining a new IBM.TieBreaker resource using the standard RMC management command **mkrsrc**.

Attention: When defining tie breaker resources, be aware that the disk on which IBM.Tiebreaker resources are stored should not also be used to store file systems.

When defining a tie breaker, you need to first determine which persistent attributes are required when defining an IBM.TieBreaker resource. This information can be returned by issuing the **mkrsrc** command with its **-e** command-line flag. The **-e** flag causes the **mkrsrc** command to display two examples of suitable command-line

input when defining a given resource. One example shows the suitable command-line input for required attributes only. The other example shows the suitable command-line input for both required and optional attributes. For example:

```
# mkrsrc -e IBM.TieBreaker
Sample mkrsrc command with required attributes:
mkrsrc IBM.TieBreaker Type=char_ptr Name=char_ptr
```

```
Sample mkrsrc command with required and optional attributes:
mkrsrc IBM.TieBreaker Type=char_ptr Name=char_ptr ReprobeData=char_ptr PreReserv
eWaitTime=uint32 DeviceInfo=char_ptr NodeInfo=sd_ptr_array PostReserveWaitTime=u
int32 HeartbeatPeriod=uint32 ReleaseRetryPeriod=uint32
```

To obtain more information on any of the attributes of an `IBM.TieBreaker` resource, use the `lsrsrcdef` command (as described in “Displaying attribute definition information for a resource or a resource class” on page 83). Here, however, we will focus only on the two that are required for defining an `IBM.TieBreaker` resource — the `Type` and `Name` attributes.

- The `Type` attribute is the name of one of the available tie-breaker types. The available tie breaker types will depend on your operating system and machine architecture. Possible types are:

Operator

This type of tie breaker asks for a decision from the system operator or administrator. The operator executes his decision by invoking the `ResolveOpQuorumTie` action as described in “Explicitly resolving a tie when the active tie-breaker type is “Operator”” on page 59.

Fail This pseudo tie breaker type always fails to reserve the tie breaker.

ECKD This tie breaker type is specific to Linux for zSeries. This tie breaker type assumes that an ECKD-DASD is shared by all nodes of the cluster. Tie breaker reservation is done by the ECKD reserve command. If creating a tie breaker of this type, you need to set the `DeviceInfo` persistent resource attribute to indicate the ECKD device number. See “Creating an ECKD tie breaker” on page 56 for more information.

SCSI This tie breaker type is specific to Linux for xSeries. This tie breaker type assumes that an SCSI-disk is shared by one or more nodes of the peer domain. Tie breaker reservation is done by the SCSI reserve or persistent reserve command. If creating a tie breaker of this type, you need to set the `DeviceInfo` persistent resource attribute to identify the SCSI device. See “Creating an SCSI tie breaker” on page 57 for more information.

DISK This tie breaker type is specific to AIX. This tie breaker type enables you to specify a SCSI or SCSI-like physical disk using an AIX device name, and assumes that the SCSI disk is shared by one or more nodes of the peer domain. Tie breaker reservation is done by the SCSI reserve or persistent reserve command. If creating a tie breaker of this type, you need to set the `DeviceInfo` persistent resource attribute to identify the physical disk. Only SCSI and SCSI-like physical disks are supported. Physical disks attached via Fiber Channel, iSCSI and Serial Storage Architecture Connections are suitable.

EXEC A script provided by an RSCT exploiter product resolves the tie situation.

The tie breaker types that are available for your operating system and machine architecture are listed in the `AvailableTypes` class attribute of the `IBM.TieBreaker` resource class. To list the available tie breaker types, you would use the `lsrsrc` command with its `-c` flag.

```
# lsrsrc -c IBM.TieBreaker AvailableTypes
Resource Class Persistent and Dynamic Attributes for: IBM.TieBreaker
resource 1:
    AvailableTypes = {"Operator",""},{"Fail",""}
```

If the **lsrsrc** command example shown above is issued on a Linux zSeries machine, the output would show ECKD as one of the available types. If issued on a Linux xSeries machine, the output would show SCSI as an available type. If issued on an AIX machine, the output would show DISK as an available type.

- The Name attribute is simply a null-terminated string you will use to identify this tie breaker. It is the value you will use when setting the OpQuorumTieBreaker persistent class attribute of the IBM.PeerNode resource class to activate the tie breaker. See “Setting the active tie breaker” on page 52 for more information.

Once you understand the values you want to assign to the persistent attributes that are required for define (and any attributes that are optional for define that you want to specify), you define the IBM.TieBreaker resource using the **mkrsrc** command. For example:

```
mkrsrc IBM.TieBreaker Name=OpQuorumTieBreaker Type=Operator
```

For complete syntax information on the **lsrsrccdef**, **lsrsrc** and **mkrsrc** commands, refer to their man pages in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating an ECKD tie breaker: The ECKD tie-breaker type is specific to Linux on zSeries. If you want to create an ECKD tie breaker object, you need to set the DeviceInfo persistent resource attribute to indicate the ECKD device number. This type of tie breaker uses a reserve/release mechanism and needs to be re-reserved periodically to hold the reservation. For this reason, we strongly recommend that you also specify the HeartbeatPeriod persistent resource attribute when creating a tie breaker of this type. The HeartbeatPeriod persistent resource attribute defines the interval at which the reservation is retried.

Attention: When defining tie breaker resources, be aware that the disk on which IBM.Tiebreaker resources are stored should not also be used to store file systems.

| If you are using the SUSE LINUX Enterprise Server 9 (SLES 9), you can obtain the
| ECKD device number by entering the command:

```
| /sbin/lscsd
```

| Output similar to the following is displayed. In the following example output, **bold**
| text is used to highlight the ECKD device number.

```
| 0.0.0100(ECKD) at ( 94: 0) is dasda : active at blocksize: 4096, 601020 blocks, 2347 MB
| 0.0.0101(FBA ) at ( 94: 4) is dasdb : active at blocksize: 512, 2454165 blocks, 1198 MB
```

| For other Linux distributions, you can obtain the device number by entering the
| command:

```
cat /proc/dasd/devices
```

| Output similar to the following is displayed. In the following example output, **bold**
| text is used to highlight the ECKD device number.

```
50dc(ECKD) at ( 94: 0) is : active at blocksize: 4096, 601020 blocks, 2347 MB
50dd(ECKD) at ( 94: 4) is : active at blocksize: 4096, 601020 blocks, 2347 MB
50de(ECKD) at ( 94: 8) is : active at blocksize: 4096, 601020 blocks, 2347 MB
50df(ECKD) at ( 94: 12) is : active at blocksize: 4096, 601020 blocks, 2347 MB
```

Once you know the device number, you can issue the **mkrsrc** command.

```
mkrsrc IBM.TieBreaker Name=eckdtest Type=ECKD DeviceInfo="ID=50dc" \  
HeartbeatPeriod=30
```

Creating an SCSI tie breaker: The SCSI tie-breaker type is specific to Linux on xSeries. If you want to create a SCSI tie breaker object, you need to specify the SCSI device using the DeviceInfo persistent resource attribute. If the SCSI configuration is different between nodes, you can also use the NodeInfo persistent resource attribute to reflect those differences.

This type of tie breaker uses a reserve/release mechanism and needs to be re-reserved periodically to hold the reservation. For this reason, we strongly recommend that you also specify the HeartbeatPeriod persistent resource attribute when creating a tie breaker of this type. The HeartbeatPeriod persistent resource attribute defines the interval at which the reservation is retried.

Attention: When defining tie breaker resources, be aware that the disk on which IBM.Tiebreaker resources are stored should not also be used to store file systems.

To obtain the identifiers for a SCSI device, enter:

```
cat /proc/scsi/scsi
```

Output similar to the following is displayed:

```
Attached devices:  
Host: scsi0 Channel: 00 Id: 00 Lun: 00  
Vendor: IBM      Model: DRVS18D      Rev: 0380  
Type: Direct-Access      ANSI SCSI revision: 03  
Host: scsi0 Channel: 00 Id: 01 Lun: 00  
Vendor: IBM      Model: DRVS18D      Rev: 0380  
Type: Direct-Access      ANSI SCSI revision: 03  
Host: scsi0 Channel: 00 Id: 15 Lun: 00  
Vendor: IBM      Model: 2104-TL1      Rev: BP18  
Type: Enclosure      ANSI SCSI revision: 03
```

Once you know the identifiers for the SCSI device, you can issue the **mkrsrc** command. If the SCSI configuration is the same on all nodes, you identify the SCSI device using the DeviceInfo persistent resource attribute. For example:

```
mkrsrc IBM.TieBreaker Name=scsi Type=SCSI DeviceInfo="ID=4 LUN=0 HOST=0,CHAN=0" \  
HeartbeatPeriod=30
```

Because the SCSI configuration can be different between nodes (even if the target device is the same), you may need to reflect differences between nodes using the NodeInfo persistent resource attribute. For example, say a SCSI device is connected to two nodes and has the following SCSI identifiers:

```
node1: HOST=0 CHAN=0 ID=4 LUN=0  
node2: HOST=1 CHAN=2 ID=4 LUN=0
```

You would create the tie breaker object by entering the following **mkrsrc** command:

```
mkrsrc IBM.TieBreaker Name=scsi Type=SCSI DeviceInfo="ID=4 LUN=0" \  
NodeInfo='{"node1", "HOST=0,CHAN=0"}, {"node2", "HOST=1 CHAN=2"}' \  
HeartbeatPeriod=30
```

For each node, the configuration resource manager merges the DeviceInfo string with the NodeInfo string. In the preceding example, the merged string for "node1" will be "ID=4 LUN=0 HOST=0 CHAN=0". Any duplicate keywords specified in the DeviceInfo and NodeInfo strings are allowed, and the last one will be used. So the preceding command could also have been specified as:

```
mkrsrc IBM.TieBreaker Name=scsi Type=SCSI DeviceInfo="ID=4 LUN=0 HOST=0,CHAN=0" \
NodeInfo='["node2", "HOST=1 CHAN=2"]}' HeartbeatPeriod=30
```

This simplification can be useful when the SCSI identifiers are the same for many nodes. You will only have to use the NodeInfo attribute to specify the nodes that are different.

Creating a DISK tie breaker: The DISK tie-breaker type is specific to AIX. If you want to create a DISK tie breaker object, you need to set the DeviceInfo persistent resource attribute to indicate the AIX device name. The AIX device name must specify a SCSI or SCSI-like physical disk that is shared by all nodes of the peer domain. Physical disks attached via Fiber Channel, iSCSI, and Serial Storage Architecture may serve as a DISK tie breaker. However, IDE hard disks do not support the SCSI protocol and cannot serve as a DISK tie-breaker. Logical volumes also cannot serve as a DISK tie breaker.

This type of tie breaker uses a reserve/release mechanism and needs to be re-reserved periodically to hold the reservation. For this reason, we strongly recommend that you also specify the HeartbeatPeriod persistent resource attribute when creating a tie breaker of this type. The HeartbeatPeriod persistent resource attribute defines the interval at which the reservation is retried.

Attention: When defining tie breaker resources, be aware that the disk on which IBM.Tiebreaker resources are stored should not also be used to store file systems.

To print every known physical volume in the system along with its physical disk name, enter the **lspv** command:

```
lspv
```

Output similar to the following is displayed:

hdisk0	000000371e5766b8	rootvg	active
hdisk1	000069683404ed54	None	

To verify that a disk is a SCSI or SCSI-like disk and so a suitable candidate for a DISK tie breaker, use the **lsdev** command. For example:

```
lsdev -C -l hdisk0
```

Output similar to the following is displayed

```
hdisk0 Available 10-60-00-0,0 16 Bit SCSI Disk Drive
```

In order to serve as a tie-breaker disk, the disk must be shared by all nodes of the peer domain. Check the physical volume ID returned by the **lspv** command to determine if the disk is shared between nodes (in the preceding output for the **lspv** command, the physical volume ID is listed in the second column; the volume ID for *hdisk0* is *000000371e5766b8*.) Be aware, however, that AIX remembers all disks that have been attached to the system, and the disks listed by the **lspv** command may no longer be attached. If such a disk was moved to another machine, it might appear that the disk is shared, when in fact it is no longer attached to the original machine.

The disk on which IBM.Tiebreaker resources are stored should not also be used to store file systems. If the nodes of the cluster share more than one disk, it may be difficult to determine which one is the tie-breaker disk, and which one is used for regular data. The output from the **lsdev** command shows the SCSI address associated with the disk. (In the preceding output for the **lsdev** command, the SCSI

address is listed in the third column; the SCSI address for *hdisk0* is *10-60-00-0,0*.) This information will help you identify the correct disk if you are aware of the disk's address prior to its installation.

Once you know the device name, you can issue the **mkrsrc** command.

```
mkrsrc IBM.TieBreaker Name=disktb Type=DISK DeviceInfo="DEVICE=/dev/hdisk0" \  
HeartbeatPeriod=30
```

Creating an EXEC tie breaker: You can only create a tie breaker of this type if an RSCT exploiter product provides a script or executable designed to resolve a tie situation. If an exploiter product has provided such a script or executable, and it has been installed on all nodes of your cluster, you can create an EXEC tie breaker object using the **mkrsrc** command. For the EXEC tie breaker type, the DeviceInfo attribute should specify the path to the script or executable and any program arguments. For example, to create the tie breaker named **MyTB** when the provided executable is **/usr/sbin/rsct/bin/tiebreaker**, you would enter:

```
mkrsrc -c IBM.TieBreaker Type="EXEC" Name="MyTB" \  
DeviceInfo='PATHNAME=/usr/sbin/rsct/bin/tiebreaker myArg=123'
```

Explicitly resolving a tie when the active tie-breaker type is "Operator"

When the active tie breaker is the predefined tie breaker "Operator" or a tie breaker whose persistent attribute Type is "Operator", then the configuration resource manager will not automatically resolve tie situations. If domain partitioning occurs with a sub-domain containing exactly half the defined nodes (or if exactly half of the domain's defined nodes become inaccessible), the configuration manager will set the OpQuorumState dynamic attribute of the PeerDomain resource to 1 (PendingQuorum). Operational quorum will not be granted until either the network is repaired, failing nodes are brought online, or you explicitly break the tie by issuing the ResolveOpQuorumTie action of the IBM.PeerNode resource class.

To resolve a tie situation using the ResolveOpQuorumTie action, you must invoke the action on a node of each active sub-domain. The single input parameter to this action is an integer that indicates whether the sub-domain in which the action is invoked is denied (0) or granted (1) or ownership of the tie breaker.

When explicitly resolving a tie between sub-domains, you should, in order to avoid corruption of shared data, first deny ownership of the tie breaker to the appropriate sub-domain. Once you have denied ownership of the tie breaker to the appropriate sub-domain, you can safely grant ownership of the tie breaker to the sub-domain that you want to have operational quorum.

To deny ownership of the "Operator" tie breaker to a sub-domain, invoke the following action on a node of that sub-domain.

```
runact -c IBM.PeerDomain ResolveOpQuorumTie Ownership=0
```

Denying ownership of the tie breaker to a sub-domain will cause the configuration manager to set the OpQuorumState dynamic attribute of the PeerDomain resource to 2 (NoQuorum). The sub-domain will lose quorum, which may in turn cause the critical resource protection method to be invoked on any nodes that have critical resources active. See "Setting the critical resource protection method for a peer domain or a node in a peer domain" on page 49 for more information.

To grant ownership of the "Operator" tie breaker to a sub-domain, invoke the following action on a node of that sub-domain.

```
runact -c IBM.PeerDomain ResolveOpQuorumTie Ownership=1
```

Granting ownership of the tie breaker to a sub-domain will cause the configuration manager to set the OpQuorumState dynamic attribute of the PeerDomain resource to 0 (HasQuorum). The sub-domain will have operational quorum and so will become the peer domain.

Chapter 4. Managing and monitoring resources using RMC and resource managers

Note: Most of the predefined conditions described in this chapter are not available in the Linux implementation of RSCT. However, these same conditions are easily created by following the instructions in “Creating a condition” on page 101.

The Resource Monitoring and Control (RMC) subsystem is the scalable backbone of RSCT that provides a generalized framework for managing and monitoring resources (physical or logical system entities) within a single system or a cluster. RMC is a daemon that runs on individual systems or each node of a cluster. It provides a single management/monitoring infrastructure for individual machines, peer domains, and management domains. RMC, however, is a generalized framework — it provides an abstract way of representing resources of a system, but it does not itself represent the actual resources. The actual resources are represented by resource managers. A resource manager is a daemon process that maps RMC’s resource abstractions into actual descriptions of resources. Since the various resource managers all define resources according to the same abstraction defined by RMC, RMC is able to manage the resources generically.

This chapter contains the following sections:

- “Understanding RMC and resource managers” on page 62 describes some key concepts you should understand before using the RMC and resource manager commands described in this chapter.
- “Managing user access to resources using RMC ACL files” on page 77 describes how to grant users the permissions they need to use RMC and the resource managers effectively.
- “Basic resource monitoring” on page 84 describes how you can use the Event Response Resource Manager to monitor resources for conditions or interest, and, should the conditions occur, respond in a specific way. This section describes how to do this using predefined conditions and responses we provide. The conditions are resource attribute thresholds that will trigger an associated response. The responses are descriptions of specific actions RMC should take when an associated condition occurs.
- “Advanced resource monitoring” on page 98 continues our discussion of using the Event Response Resource Manager to respond in an event-driven way to system conditions. While “Basic resource monitoring” on page 84 describes how to do this using predefined conditions and responses that we provide, this section describes how to create your own conditions and responses. It also describes how to extend RMC monitoring/response capabilities by defining sensors and response scripts. A sensor is a command that the RMC runs (at specified intervals and/or when you explicitly request for it to be run) to retrieve one or more user-defined values. These values are your own defined attributes and can be used as part of a condition you define. A response script is a script that defines how the system should react to a particular condition and can be used as part of a response you define.
- “Using expressions to specify condition events and command selection strings” on page 130 provides detailed information on how to create event expressions and selection string expressions. An event expression is defined as part of a condition; RMC tests the event expression periodically to determine if the condition is true. Selection string expressions, on the other hand, can be specified on a number of RMC and resource manager commands discussed in

this chapter, and are used to restrict the commands' actions in some way. For example, a selection string expression could identify a subset of resources for a command to act upon. While creating expressions is a fairly intuitive task (the expressions are similar to a C language statement or WHERE clause of an SQL query), this section provides reference information on supported types, operators, and so on.

Understanding RMC and resource managers

This section describes some key concepts you need to understand before performing the various tasks outlined in this chapter. It describes:

- how the RMC subsystem provides a generic way to represent, and manage various physical and logical system entities.
- how a set of resource managers map information about specific entities to RMC's abstractions.
- the representational components of RMC's generic framework. These include resources (the physical or logical system entities represented), attributes (characteristics of resources), and resource classes (sets of resources with common attributes).
- the resource managing capabilities of RMC and the resource managers.
- the monitoring capabilities of RMC and the resource managers (described in more detail later in "Basic resource monitoring" on page 84 and "Advanced resource monitoring" on page 98).
- how RMC implements authorization (described in more detail later in "Managing user access to resources using RMC ACL files" on page 77).
- differences between using RMC on a single node versus a cluster.

What is RMC?

The Resource Monitoring and Control (RMC) is a generalized framework for managing, monitoring, and manipulating resources (physical or logical system entities). RMC runs as a daemon process on individual machines, and, therefore, is scalable. You can use it to manage and monitor the resources of a single machine, or you can use it to manage and monitor the resources of a cluster's peer domain or management domain. In a peer domain or management domain, the RMC daemons on the various nodes work together to enable you to manage and monitor the domain's resources.

What is a resource?

A *resource* is the fundamental concept of RMC's architecture. It refers to an instance of a physical or logical entity that provides services to some other component of the system. The term resource is used very broadly to refer to software as well as hardware entities. For example, a resource could be a particular file system or a particular host machine.

What is a resource class?

A *resource class* is a set of resources of the same type. For example, while a resource might be a particular file system or particular host machine, a resource class would be the set of file systems, or the set of host machines. A resource class defines the common characteristics that instances of the resource class can have; for example, all file systems will have identifying characteristics (such as a name), as well as changing characteristics (such as whether or not it is mounted). Each individual resource instance of the resource class will then define what its particular characteristic values are (for example, this file system is named `"/var"`, and it is currently a mounted file system).

What are resource attributes?

A resource *attribute* describes some characteristic of a resource. If the resource represents a host machine, its attributes would identify such information as the host name, size of its physical memory, machine type, and so on.

What is the difference between persistent attributes and dynamic attributes?:

There are two types of resource attributes — *persistent attributes* and *dynamic attributes*. The attributes of a host machine just mentioned (host name, size of physical memory, and machine type) are examples of *persistent attributes* — they describe enduring characteristics of the resource. While you could change the host name or increase the size of its physical memory, these characteristics are, in general, stable and unchanging. *Dynamic attributes*, on the other hand, represent changing characteristics of the resource. Dynamic attributes of a host resource, for example, would identify such things as the average number of processes that are waiting in the run queue, processor idle time, the number of users currently logged on, and so on.

Persistent attributes are useful for identifying particular resources of a resource class. In this chapter, we discuss many commands for directly or indirectly manipulating resources. Persistent attributes enable you to easily identify an individual resource or set of resources of a resource class that you want to manipulate. For example, the **lsrsrc** command lists resource information. By default, this command will list the information for all resources of the class. However, you can filter the command using persistent attribute values. In a cluster, this ability would enable you to list information about a particular host machine (by filtering using the host's name) or a group of host machines of the same type (by filtering according to the machine type). Although listing resources is a fairly simple task, this same ability to identify resources by their attributes, and isolate command actions to a single resource or subset of resources, is available on many of the more advanced commands described in this chapter. This ability gives you increased flexibility and power in managing resources.

Dynamic attributes are useful in monitoring your system for conditions of interest. As described in “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98, you can monitor events of interest (called *conditions*) and have the RMC system react in particular ways (called *responses*) if the event occurs. The conditions are logical expressions based on the value of an attribute. For example, there is a resource class used to represent file systems. You could create a condition to monitor the file systems and trigger a response if any of them become more than 90 percent full. The percentage of space used by a file system is one of its dynamic attribute values. It usually does not make sense to monitor persistent attribute values, since they are generally unchanging. For example, if you wanted to monitor a file system, it would not make sense to monitor based on the file system name (a persistent attribute). However, you may want to use this persistent attribute to identify a particular file system resource to monitor. Instead of monitoring all file systems, you could use this persistent attribute value to identify one particular file system to monitor.

What is a resource manager?

A resource manager is a daemon process that provides the interface between RMC and actual physical or logical entities. It is important to understand that although RMC provides the basic abstractions (resource classes, resources, and attributes) for representing physical or logical entities, it does not itself represent any actual entities. A resource manager maps actual entities to RMC's abstractions.

Each resource manager represents a specific set of administrative tasks or system features. The resource manager identifies the key physical or logical entity types related to that set of administrative tasks or system features, and defines resource classes to represent those entity types.

For example, the Host resource manager contains a set of resource classes for representing aspects of a individual host machine. It defines resource classes to represent

- individual machines (IBM.Host)
- paging devices (IBM.PagingDevice)
- physical volumes (IBM.PhysicalVolume)
- processors (IBM.Processor)
- a host's identifier token (IBM.HostPublic)
- programs running on the host (IBM.Program)
- each type of adapter supported by the host, including ATM adapters (IBM.ATMDevice), Ethernet adapters (IBM.EthernetDevice), FDDI adapters (IBM.FDDIDevice), and token-ring adapters (IBM.TokenRingDevice)

The resource class definitions describe the persistent and dynamic attributes that individual resource instances of that class can or must define. For example, the Host resource class defines persistent attributes such as Name (the name of the host machine), RealMemSize (the size of physical memory in bytes), and OsVersion (the version of the operating system or kernel running on the host machine). It defines dynamic attributes such as PctTotalTimeIdle (system-wide percentage of time that processors are idle), NumUsers (number of users currently logged on to the system), and UpTime (the number of seconds since the system was last booted).

A resource manager also determines how individual resources of each class are identified. Although you can use the **mkrsrc** command to explicitly define a resource, this is often not necessary, since resources may be automatically harvested by the resource manager. For example, there is resource manager used to represent file systems. This resource manager harvests (gathers information on) existing file systems to create resources representing those file systems. It will periodically repeat this harvesting to so that its resources are still representative of the actual file systems available. In addition to harvesting, resources may be created implicitly by other commands. For example, the Host resource manager has a Program resource class that represents programs running on the host. If you were to create a monitoring condition (described in “Creating a condition” on page 101) referring to a particular program, a Program resource representing the program is created implicitly.

Another job of a resource manager to to determine the dynamic attribute values of its resources. Since dynamic attributes represent changing characteristics of a resource, the resource manager will periodically poll the actual resources to determine the dynamic attribute values. This is essential to enable resource monitoring (described in “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98) where conditions used to trigger responses are logical expressions based on the value of an attribute. It is the periodic polling of resources that enables the event driven condition/response behavior of resource monitoring.

While some resource managers represent system features (such as individual host machines of a cluster, or file systems) other represent resources related to a

specific administrative task (such as peer domain configuration, or resource monitoring). Since the purpose of such a resource manager is to provide administrative function, it will provide a command-line interface for performing the administrative tasks. For example, the Configuration resource manager (described in Chapter 3, “Creating and administering an RSCT peer domain,” on page 19) provides commands for creating creating a peer domain, adding nodes to the domain, taking the domain offline, and so on.

Each resource manager has a startup mode that determines when the RMC subsystem will start it. The three startup modes are:

auto-start

The resource manager is started when the RMC subsystem is started.

on-line auto-start

The resource manager is started when the node becomes online in a peer domain.

on-demand

The resource manager is started when one of its resources is first referenced.

The startup mode for each RSCT resource manager is shown in Table 8.

What resource managers are provided with RSCT?

The following resource managers are provided as part of RSCT. Together with the RMC subsystem, they provide the administrative and monitoring capabilities of RSCT. Keep in mind that additional resource managers are provided by certain cluster licensed program products (such as CSM, which contains the Domain Management resource manager).

Table 8. Resource Managers Provided with RSCT

Resource manager:	Description:	Startup Mode
Audit log resource manager	Provides a system-wide facility for recording information about the system’s operation. It is use by subsystem components to log information about their actions, errors, and so on. In particular, the Event Response resource manager, which contains the resource monitoring functionality, uses the audit log resource manager to log information about condition events occurring, what responses were taken, and so on. A command-line interface to the audit log resource manager enables you to list and remove records from and audit log. For more information on the audit log resource manager’s commands, refer to “Using the audit log to track monitoring activity” on page 93. Complete syntax information on the commands is provided in the <i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> . For more information, refer to “What resource classes are defined by the audit log resource manager?” on page 66.	on-demand
CIM resource manager	Enables you to use RMC to query system configuration through Common Information Model (CIM) classes. The CIM resource manager provides a command (mkcimreg) that enables you to register CIM classes with RMC. Once registered, you can query the value of CIM properties using the standard RMC command lsrsrc . For more information, refer to “Querying CIM properties” on page 122. Complete syntax information on the mkcimreg command is provided in the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> .	on-demand
Configuration resource manager	Provides the ability, through its command-line interface, to create and administer a peer domain (a cluster of nodes configured for high availability). For more information on the configuration resource manager’s commands, refer to Chapter 3, “Creating and administering an RSCT peer domain,” on page 19. Complete syntax information on the commands is provided in the <i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> . For more information, refer to “What resource classes are defined by the configuration resource manager?” on page 67.	on-demand (if offline). Otherwise on-line auto-start

Table 8. Resource Managers Provided with RSCT (continued)

Resource manager:	Description:	Startup Mode
Event response resource manager	Provides resource monitoring — the ability to take actions in response to conditions occurring in the system. Its command-line interface enables you to associate conditions with responses, start and stop condition monitoring, and so on. For more information on the event response resource manager's commands, refer to "Basic resource monitoring" on page 84 and "Advanced resource monitoring" on page 98. Complete syntax information on the commands is provided in the <i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> . For more information, refer to "What resource classes are defined by the event response resource manager?" on page 68.	auto-start
File system resource manager	Provides a resource class to represent file systems. This resource manager has no user interface. Instead, you interact with it indirectly when you monitor its resource attributes using the event response resource manager. For more information, refer to "What resource classes are defined by the file system resource manager?" on page 68.	on-demand
Host resource manager	Provides resource classes to represent an individual machine, including its paging devices, physical volumes, and processors. This resource manager has no user interface. Instead, you interact with it indirectly when you monitor its resource attributes using the event response resource manager. For more information, refer to "What resource classes are defined by the host resource manager?" on page 68.	on-demand
Least-privilege resource manager	Controls access to root commands or scripts, and local or remote execution of those commands or scripts on AIX or Linux. The least-privilege (LP) resource manager provides a resource class and a command-line interface that allow you to define, manage, and monitor root commands and scripts as LP resources. For more information about the LP resource manager, refer to "What resource classes are defined by the least-privilege resource manager?" on page 69 and Chapter 5, "Controlling access to root commands and scripts," on page 141.	on-demand
Sensor resource manager	Provides a way to extend the monitoring capabilities of the system by enabling you to create a single user-defined attribute for monitoring. Extending the system in this way involves creating a <i>sensor</i> . A sensor is merely a command that the RMC subsystem runs (at specified intervals and/or when you explicit request for it to be run) to retrieve one or more user-defined values. The sensor is essentially a resource that you add to the Sensor resource class of the Sensor resource manager. The values returned by the script are dynamic attributes of that resource. Using the event response resource manager commands, you can then create a condition to monitor any of the attributes you have defined. The sensor resource manager provides a command-line interface for creating, changing, listing, and removing sensors. For more information on the sensor resource manager's commands, refer to "Creating event sensor commands for monitoring" on page 109. Complete syntax information on the commands is provided in the <i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> . For more information, refer to "What resource classes are defined by the sensor resource manager?" on page 69.	on-demand

What resource classes are defined by the audit log resource manager?: The following table describes the resource classes defined by the audit log resource manager. In general, you will not need to manipulate resources of these classes directly. Instead you would manipulate the audit log using the **lsaudrec** command (as described in "Using the audit log to track monitoring activity" on page 93) and **rmaudrec** command (as described in "Deleting entries from the audit log using the rmaudrec command" on page 96). One instance where you would manipulate a resource of the IBM.AuditLog class directly would be to set the RetentionPeriod or MaxSize attribute values (as described in "Deleting entries from the audit log using the IBM.AuditLog resource's RetentionPeriod and MaxSize attributes" on page 97).

Table 9. Resource classes defined by the audit log resource manager

Resource class:	Description:
IBM.AuditLog	Each resource of this class represents a subsystem that can add records to the audit log.
IBM.AuditLogTemplate	Each resource of this class represents a template for an audit log record.

For information on listing attribute values or attribute definitions for the resource classes (or resource instances of the resource classes) listed in the preceding table, refer to “Listing Resource Information” on page 79.

What resource classes are defined by the CIM resource manager?: The CIM resource manager does not provide a default set of resource classes. Instead, the CIM resource manager enables you to use RMC to query system configuration through Common Information Model (CIM) classes. CIM is a data model, similar to the RMC data model, for organizing computer hardware and software resources into a common object-oriented class hierarchy.

The CIM resource manager provides a command that enables you to register CIM properties with RMC. The CIM classes are mapped to new RMC resource classes. The RMC resource class name will be a concatenation of the namespace and the CIM class name — for example *cimv2.Linux_ComputerSystem*. All registered CIM classes are placed in the root/cimv2 namespace.

For more information on the CIM resource manager, refer to “Querying CIM properties” on page 122.

What resource classes are defined by the configuration resource manager?:

The following table describes the resource classes defined by the configuration resource manager. In general, you will not need to manipulate resources of these classes directly. Instead you would use the configuration resource manager commands described in Chapter 3, “Creating and administering an RSCT peer domain,” on page 19. However, you may need to:

- modify attributes of the RSCTParameters class (as described in “Modifying Topology Services and Group Services parameters” on page 46).
- modify the CritRsrcProtMethod attribute of the IBM.PeerNode class or and resource instance of the IBM.PeerNode class (as described in “Setting the critical resource protection method for a peer domain or a node in a peer domain” on page 49).
- modify the OpQuorumOverride attribute of the IBM.PeerNode class (as described in “Overriding the configuration resource manager’s operational quorum calculation to force operational quorum” on page 51).
- modify the OpQuorumTieBreaker attribute of the IBM.PeerNode class (as described in “Setting the active tie breaker” on page 52).

Table 10. Resource classes defined by the configuration resource manager

Resource class:	Description:
IBM.CommunicationGroup	Each resource of this class represents a communication resource upon which liveness checks (Topology Services “heartbeating”) will be performed.
IBM.NetworkInterface	Each resource of this class represents an IP network interface that exists in the peer domain.
IBM.PeerDomain	Each resource of this class represents an RSCT peer domain in which a particular node is defined. Each node has its own IBM.PeerDomain resource class, with each instance of the resource class representing an RSCT peer domain in which the node is defined. The number of instances of this resource class, therefore, indicates the number of peer domains in which the node is defined.
IBM.PeerNode	Each resource of this class represents a node defined in the peer domain. A node is here defined as an instance of an operating system, and is not necessarily tied to hardware boundaries.
IBM.RSCTParameters	Represents operational characteristics of RSCT subsystems.

Table 10. Resource classes defined by the configuration resource manager (continued)

Resource class:	Description:
IBM.TieBreaker	Each resource of this class represents a tie-breaker. A tie-breaker is used, when domain partitioning results in two sub-domains each containing exactly half the nodes, to determine which sub-domain has operational quorum.

For information on listing attribute values or attribute definitions for the resource classes (or resource instances of the resource classes) listed in the preceding table, refer to “Listing Resource Information” on page 79.

What resource classes are defined by the event response resource manager?: The following table describes the resource classes defined by the event response resource manager. In general, you will not need to manipulate resources of these classes directly. Instead you would use the event response resource manager commands described in “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98.

Table 11. Resource classes defined by the event response resource manager

Resource class:	Description:
IBM.Association	Each resource of this class represents a condition/response association. Such an association enables the RMC subsystem to trigger one or more response actions when a particular condition occurs.
IBM.Condition	Each resource of this class represents a condition (an event that should trigger a response).
IBM.EventResponse	Each resource of this class represents a response (one or more actions the system can take when a condition event occurs).

For information on listing attribute values or attribute definitions for the resource classes (or resource instances of the resource classes) listed in the preceding table, refer to “Listing Resource Information” on page 79.

What resource classes are defined by the file system resource manager?: The following table describes the resource class defined by the file system resource manager. In general, you will not need to manipulate resources of this class directly. However, you may want to monitor file systems using the dynamic attributes of IBM.FileSystem resources. Monitoring is described in “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98.

Table 12. Resource classes defined by the file system resource manager

Resource class:	Description:
IBM.FileSystem	Each resource of this class represents a filesystem.

For information on listing attribute values or attribute definitions for the resource class (or resource instances of the resource classes) shown in the preceding table, refer to “Listing Resource Information” on page 79.

What resource classes are defined by the host resource manager?: The following table describes the resource classes defined by the host resource manager. In general, you will not need to manipulate resources of these classes directly. However, you may want to monitor host machines using resource dynamic attributes of the various resource classes. Monitoring is described in “Basic resource monitoring” on page 84 and “Advanced resource monitoring” on page 98.

Table 13. Resource classes defined by the host resource manager

Resource class:	Description:
IBM.ATMDevice	Each resource of this class represents an ATM adapter installed on the host.
IBM.EthernetDevice	Each resource of this class represents an Ethernet adapter installed on the host.
IBM.FDDIDevice	Each resource of this class represents a FDDI adapter installed on the host
IBM.Host	Each resource of this class represents a host machine that is running a single copy of an operating system.
IBM.HostPublic	Each resource of this class represents the host's public key.
IBM.PagingDevice	Each resource of this class represents a device that is used by the operating system for paging.
IBM.PhysicalVolume	Each resource of this class represents a physical volume.
IBM.Processor	Each resource of this class represents a processor.
IBM.Program	Each resource of this class represents a program that is running on the host.
IBM.TokenRingDevice	Each resource of this class represents a token-ring device installed on the host.

For information on listing attribute values or attribute definitions for the resource classes (or resource instances of the resource classes) listed in the preceding table, refer to "Listing Resource Information" on page 79.

What resource classes are defined by the least-privilege resource manager?:

The following table describes the resource class defined by the least-privilege resource manager. In general, you will not need to manipulate resources of this class directly. Instead you will use the least-privilege resource manager commands described in Chapter 5, "Controlling access to root commands and scripts," on page 141.

Table 14. Resource classes defined by the least-privilege resource manager

Resource class:	Description:
IBM.LPCCommands	Each resource of this class represents a root command or script that only authorized users may run.

For information on listing attribute values or attribute definitions for the resource class (or resource instances of the resource classes) shown in the preceding table, refer to "Listing Resource Information" on page 79.

What resource classes are defined by the sensor resource manager?: The following table describes the resource class defined by the sensor resource manager. In general, you will not need to manipulate resources of this class directly. Instead you will use the sensor resource manager commands described in "Creating event sensor commands for monitoring" on page 109.

Table 15. Resource classes defined by the sensor resource manager

Resource class:	Description:
IBM.Sensor	Each resource of this class represents a sensor (a command that the RMC runs to retrieve one or more user-defined values).

For information on listing attribute values or attribute definitions for the resource class (or resource instances of the resource classes) shown in the preceding table, refer to "Listing Resource Information" on page 79.

How does RMC and the resource managers enable you to manage resources?

As already described, RMC provides resource class abstractions for representing physical or logical system entities, while the individual resource managers map actual entities to these abstractions. Since the various resource managers all define resources according to the same abstractions defined by RMC, RMC is able to manage the resources generically. RMC provides a set of commands that enable you to list information about and manipulate resources, regardless of which resource manager defines the particular resource class.

Often these general RMC commands are not needed. For example, a **mkrsrc** command exists, enabling you to define a new resource of a particular class. However, the resource managers often automatically harvest this information to create the resources, or certain resource manager commands explicitly or implicitly create the resource. For example, the event response resource manager provides the **mkcondition** command to create a condition for resource monitoring. The **mkcondition** command creates a Condition resource; there is no need to use the generic **mkrsrc** command.

The RMC commands you will use most commonly are the **lsrsrc** and **lsrsrdef** commands which display resource or resource class information you may need when issuing other commands. The **lsrsrc** command lists the persistent and/or dynamic attributes of resources, and the **lsrsrdef** lists a resource class definition. For more information on the **lsrsrc** and **lsrsrdef** commands, refer to “Listing Resource Information” on page 79.

For complete syntax and reference information on the generic RMC commands refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*

How do RMC and the resource managers enable you to monitor resources?

RMC and the resource managers together provide sophisticated monitoring and response capabilities that enable you to detect, and in many cases correct, system resource problems such as a critical file system becoming full. You are able to monitor virtually all aspects of your system resources and specify a wide range of actions to take — from general notification or logging capabilities we provide to more targeted recovery responses you define.

The resource monitoring capability is largely provided by the event response resource manager (although you are typically monitoring dynamic attribute values provided by the host resource manager, file system resource manager, and sensor resource manager). The event response resource manager provides a set of commands that enable you to monitor events of interest (called *conditions*) and have the RMC system react in particular ways (called *responses*) if the event occurs.

What is a condition?

A *condition* specifies the event that should trigger a response. It does this using an *event expression*.

What is an event expression?: An *event expression* consists of an attribute name, a mathematical comparison symbol, and a constant. For example, the IBM.FileSystem resource class defines a dynamic attribute PercentTotUsed to

represent the percentage of space used in a file system. The following event expression, if specified on a condition, would trigger an event if a file system resource in the resource class was over 90 percent full:

```
PercentTotUsed > 90
```

The condition's event expression will, by default, apply to all resources of a particular resource class (in this example, all file systems). However, using a selection string that filters the resources based on persistent attribute values, you can create a condition that applies only to a single resource of the resource class or a subset of its resources. For example, the following selection string, if specified on a condition, would specify that the condition applies only to the **/var** file system. This selection string uses the persistent attribute Name of the resource class to identify the **/var** file system.

```
"Name == \"/var\""
```

Our condition now will now trigger an event only if the **/var** file system is over 90 percent full. When the condition is later active, RMC will periodically test the event expression at set intervals to see if it is true. If the expression does test true, RMC triggers any responses associated with the condition.

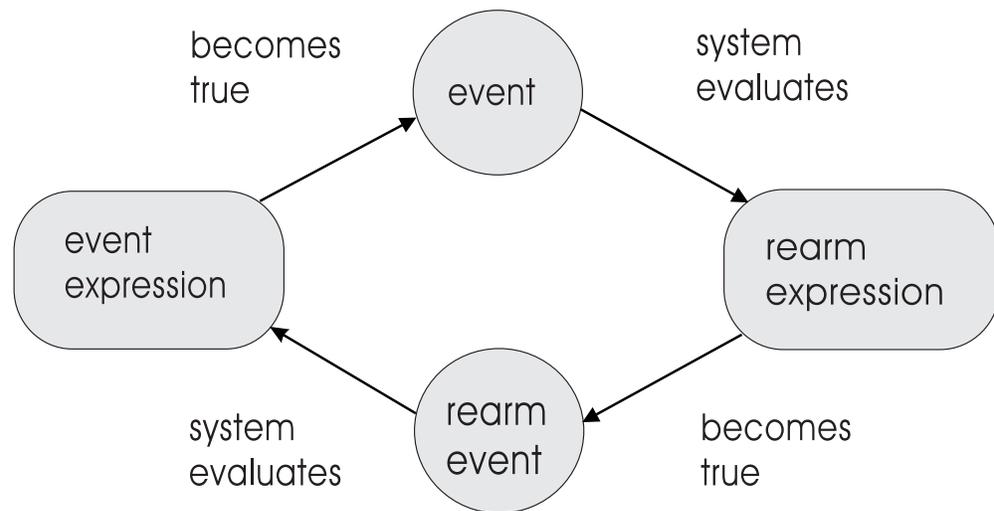
As already stated, each event expression refers to a particular attribute value (usually a dynamic attribute), which will be polled by RMC at set intervals to determine if the expression tests true. RMC keeps track of the previously observed value of the attribute, so the event expression can compare the currently observed value with the previously observed value. If the event expression suffixes the attribute name with "@P", this represents the previously observed value of the attribute. For example, the following event expression, if specified on a condition, would trigger an event if the average number of processes on the run queue has increase by 50% or more between observations.

```
(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)
```

What is a rearm event expression?: A condition can optionally have a *rearm event expression* defined. If it does, then RMC will stop evaluating the event expression once it tests true, and instead will evaluate the rearm event expression until it tests true. Once the rearm event expression tests true, the condition is rearmed. In other words, RMC will once again evaluate its event expression. For example, our event expression tests to see if the **/var** file system is 90 percent full. If it is, the associated response is triggered. We might not want RMC to continue evaluating this same expression and so triggering the same response over and over. If the response was to notify you by e-mail of the condition, the first e-mail would be enough. That's where a rearm event expression comes in. The following expression, if specified as the condition's rearm event expression, will rearm the condition once the **/var** file system in less than 75 percent full.

```
PercentTotUsed < 75
```

The following diagram illustrates the cycle of event expression/rearm event expression evaluation.



What is a condition's monitoring scope?: Another important feature of a condition is its *monitoring scope*. The *monitoring scope* refers to the node or set of nodes where the condition is monitored. Although a condition resource is defined on a single node, its monitoring scope could be the local node only, all the nodes of a peer domain, select nodes of a peer domain, all the nodes of the management domain, or select nodes of a management domain. If the monitoring scope indicates nodes of a peer domain, the node on which the condition resource is defined must be part of the peer domain. If the monitoring scope indicates nodes of a management domain, the node on which the condition resource is defined must be the management server of the management domain.

How do I create conditions?: It is important to understand that, in most cases, you will not need to create conditions since we have provided a set of predefined conditions to monitor most of the dynamic attributes defined by the file system resource manager and host resource manager. You can list these predefined conditions using the **lscondition** command described in "Listing conditions" on page 85. If the predefined conditions are not sufficient, you can create your own to monitor any attribute. To do this, you use the **mkcondition** command as described in "Creating a condition" on page 101. Even if you are creating your own conditions, you can usually copy one of our predefined ones to use as a template, modifying it as you see fit. If none of the attributes we provide contains the value you are interested in monitoring, you can extend the RMC system by creating a sensor. A *sensor* is merely a command that the RMC system runs (at specified intervals and/or when you explicitly request for it to be run) to retrieve one or more user-defined values. For more information, refer to "Creating event sensor commands for monitoring" on page 109.

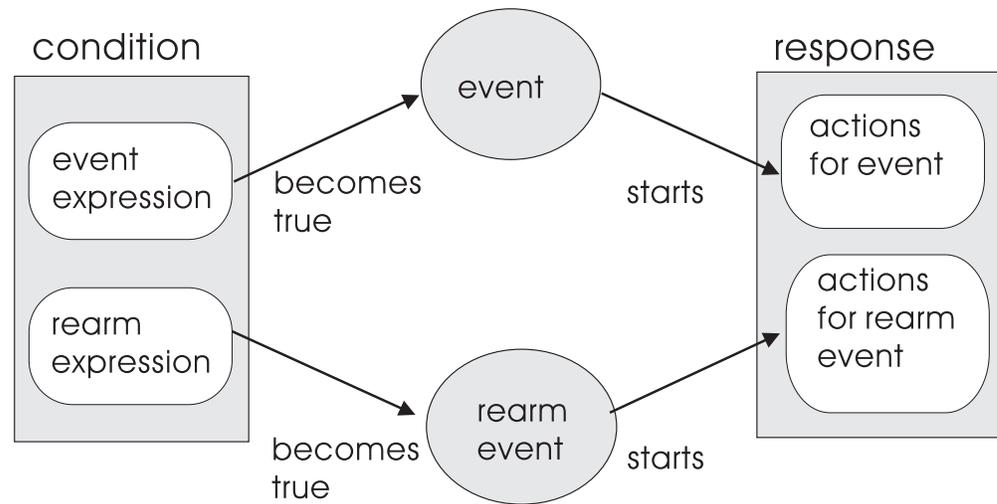
What is a response?

A *response* indicates one or more *actions* that the system can take when a condition event occurs. A *condition event* occurs when a condition's event expression or rearm event expression tests true. When such an event occurs, a response associated with the condition is triggered and any number of its *actions* can execute.

What is an action?: An *action* is simply a command or script that responds to the condition event. These response actions could perform a general-purpose action

such sending e-mail notifying you of the event, or logging the event information to a file. In fact we provide several predefined action scripts that perform such general-purpose actions. You can also write your own scripts to provide more specific responses to events. For example, if a condition tests to see if a directory is over 90 percent full, an associated response action could automatically delete the oldest unnecessary files in the directory.

A response can have multiple actions, enabling the system to respond one way to a condition event and another way to a condition rearm event (as illustrated in the following diagram).



Having multiple actions also enables a response to behave differently based on the day of the week and time of day that the event occurs. One action might be triggered on weekdays during working hours, while another might be triggered on the weekends and on weekdays outside working hours. For example, say you have a condition that will trigger an event if a processor goes offline. During working hours, you might want the system to send you e-mail when this happens. Outside work hours, the system could instead log the information to a file that you check when you come into the office.

How do I create responses?: You can think of a response as a container for one or more actions that the system can take when an associated condition event occurs. Using the **mkresponse** command (as described in “Creating a response” on page 114), you can add a single action to the response. You can then use the **chresponse** command (as described in “Modifying a response” on page 120) to add more actions to the response.

Just as we provide a set of predefined conditions you can use, we also provide a set of predefined responses. These responses utilize predefined action scripts that we also provide. The following table details these predefined responses.

Table 16. Predefined Responses

Response Name	Action(s)	Description	Action in effect:
Broadcast event on-shift	Broadcast message	Uses the predefined action script /usr/sbin/rsct/bin/wallevent to broadcast an event or rearm event to all users that log in to the host.	8AM-5PM, Monday to Friday

Table 16. Predefined Responses (continued)

Response Name	Action(s)	Description	Action in effect:
Broadcast details of event any time	Broadcast event details	Available on Linux nodes only. Uses the predefined action script /usr/sbin/rsct/bin/wallevent to broadcast an event or rearm event to all users that log in to the host. Specifies the wallevent script's -c flag to broadcast event details.	All day, everyday
"E-mail root off-shift	E-mail root	Uses the predefined action script /usr/sbin/rsct/bin/notifyevent to send an e-mail to root when an event or a rearm event occurs.	5PM-12PM, Monday to Friday 12AM-8AM, Monday to Friday All day, Saturday and Sunday
E-mail root anytime	E-mail root	Uses the predefined action script /usr/sbin/rsct/bin/notifyevent to send an e-mail to root when an event or a rearm event occurs.	All day, everyday
Log event anytime	Log event	Uses the predefined action script /usr/sbin/rsct/bin/logevent to log an entry to /tmp/systemEvents whenever an event or a rearm event occurs.	All day, everyday
Informational notifications	Log info event	Uses the predefined action script /usr/sbin/rsct/bin/logevent to log an entry to /tmp/infoEvents whenever an event or a rearm event occurs.	All day, everyday
	E-mail root	Uses the predefined action script /usr/sbin/rsct/bin/notifyevent to send an e-mail to root when an event or a rearm event occurs.	8AM-5PM, Monday to Friday
Warning notifications	Log warning event	Uses the predefined action script /usr/sbin/rsct/bin/logevent to log an entry to /tmp/warningEvents whenever an event or a rearm event occurs.	All day, everyday
	E-mail root	Uses the predefined action script /usr/sbin/rsct/bin/notifyevent to send an e-mail to root when an event or a rearm event occurs.	All day, everyday
Critical notifications	Log critical event	Uses the predefined action script /usr/sbin/rsct/bin/logevent to log an entry to /tmp/criticalEvents whenever an event or a rearm event occurs.	All day, everyday
	E-mail root	Uses the predefined action script /usr/sbin/rsct/bin/notifyevent to send an e-mail to root when an event or a rearm event occurs.	All day, everyday
	Broadcast message	Uses the predefined action script /usr/sbin/rsct/bin/wallevent to broadcast an event or rearm event to all users that log in to the host.	All day, everyday
Generate SNMP trap	SNMP trap	Uses the predefined action script /usr/sbin/rsct/bin/snmpevent to send a Simple Network Management Protocol (SNMP) trap of an ERRM event to a host running an SNMP agent.	All day, everyday

What is a condition/response association?

Before you can actually monitor a condition, you must link it with one or more responses. This is called a *condition/response association* and is required for monitoring so that RMC knows how to respond when the condition event occurs. You can create a condition/response association using either the **mkcondresp** or

startcondresp commands. The **mkcondresp** command makes the association, but does not start monitoring it. The **startcondresp** command either starts monitoring an existing association, or defines the association and starts monitoring it. For more information refer to “Creating a condition/response association” on page 88 and “Starting condition monitoring” on page 90.

What should I monitor?

To get an idea of what you can monitor, take a look at our predefined conditions. You can list the predefined conditions using the **lscondition** command (described in “Listing conditions” on page 85).

You can also create a condition based on any attribute of a resource class. Since persistent attributes are generally unchanging, it makes the most sense to monitor a dynamic attribute. You can list the dynamic attributes using the **lsrsrc** command (described in “Displaying attribute value information for a resource or a resource class” on page 81) and the **lsrsrdef** command (described in “Displaying attribute definition information for a resource or a resource class” on page 83).

Keep in mind that additional resource managers are provided by certain cluster licensed program products such as Cluster Systems Management (CSM), which provides the Domain Management Resource Manager. These additional resource managers may have resource classes with their own predefined conditions and their own attributes. Refer to the documentation for these licensed program products for details on any predefined conditions or attributes they provide.

One thing we can recommend that you monitor is the size of the **/var** file system. We recommend you do this because many RSCT subsystems make extensive use of this file system. To monitor the **/var** file system, you can use the predefined condition **/var space used** provided by the File System Resource Manager. If you are a CSM customer, you can also use the predefined condition **AnyNodeVarSpaceUsed** provided by the Domain Management Server Resource Manager. The Domain Management Server Resource Manager is only provided as part of CSM. The **AnyNodeVarSpaceUsed** condition monitors the **/var** file system on all nodes of the management domain.

How does RMC implement authorization?

RMC implements authorization using an Access Control List (ACL) file. Specifically, RMC uses the ACL file on a particular node to determine the permissions that a user must have in order to access particular resource classes and their resource instances on that node. For example, in order to modify a persistent attribute for an instance of a resource class on a particular node, the user must have write permission for that resource class on that node. To monitor an attribute, the user must have read permission. A node’s RMC ACL file is named **ctrmc.acls** and is installed in the directory **/usr/sbin/rsct/cfg**. You can have RMC use the default permissions set in this file, or you can modify it after copying it to the directory **/var/ct/cfg** as described in “Managing user access to resources using RMC ACL files” on page 77.

How do I determine the target nodes for a command?

RMC is a daemon that runs on individual systems or each node of a cluster. It provides a single management/monitoring infrastructure for individual machines, peer domains, and management domains. (For more information on domains, refer to “What are management domains and peer domains?” on page 1.) It is important for you to understand that you can execute RMC and resource manager commands on a single machine, all the nodes of a peer domain, or all the nodes of a

management domain. Some commands enable you to refine this even further, allowing you to specify a subset of nodes in the peer domain or management domain. When working in a cluster, you can also, from a local node, issue commands to be executed on another node.

There are two environment variables that, together with various command flags, determine the node(s) that will be affected by the RMC and resource manager commands you enter. These are described in the following table.

Table 17. Environment variables to determine target node(s) of a command

This environment variable:	Does this:
CT_CONTACT	Determines the system where the session with the RMC daemon occurs. When set to a host name or IP address, the command contacts the RMC daemon on the specified host. If not set, the command contacts the RMC daemon on the local system where the command is being run.
CT_MANAGEMENT_SCOPE	Identifies the management scope. The management scope determines the set of possible target nodes for the command. The default is local scope. The valid values are: <ul style="list-style-type: none"> 0 the local scope. (This is either the local machine or the machine indicated by the CT_CONTACT environment variable). 1 the local scope. (This is either the local machine or the machine indicated by the CT_CONTACT environment variable). 2 the peer domain scope. (This is either the peer domain in which the local machine is online, or the peer domain in which the machine indicated by the CT_CONTACT environment variable is online). 3 the management domain scope.

Not all of the RMC and resource manager commands use these environment variables, and the ones that do may have command-line flags you can use to override the environment variable setting or otherwise determine how the command uses the specified values.

Note for CSM Users:

When the scope is set to the management domain scope (either through the CT_MANAGEMENT_SCOPE environment variable or through command line options), RMC commands issued from the management server will return information for managed nodes. Some of these nodes may also be in peer domains within the management domain.

Certain RMC class operations return information about a node's peer domain. When performing these operations in a management domain scope, some nodes might not be in a peer domain. In these cases, the peer domain field will simply provide the local host name. When a local host name is provided instead of a peer domain name, the name will appear in angle brackets (for example: <local_host_name>).

The *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference* contain complete reference information for all of the RSCT commands. The reference information contains details on how each command uses these environment variables. The same reference information can be found for any command by viewing its online man page.

Targeting Node(s):

When this chapter discusses a command, it focuses on the command's basic function (listing condition, starting monitoring, viewing an audit log), and does not cover targeting nodes in the body of the discussion. As just described, however, many of these commands can target the local node, a

remote node, a group of nodes in a peer domain, an entire peer domain, a node in a management domain, and so on. Where appropriate, any information on how the particular command handles the targeting of nodes is covered in a separate **"Targeting Node(s)"** note like this one.

Managing user access to resources using RMC ACL files

RMC implements authorization using an access control list (ACL) file. Specifically, RMC uses the ACL file on a particular node to determine the permissions that a user must have in order to access resource classes and their resource instances. A node's RMC ACL file is named **ctrmc.acls** and is installed in the directory **/usr/sbin/rsct/cfg**. You can allow RMC to use the default permissions set in this file, or you can modify the file after copying it to the directory **/var/ct/cfg/** as described in "How to modify the ACL file" on page 79.

Format of an ACL file

An ACL file has a stanza format consisting of a stanza name followed by 0 or more stanza lines:

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

A stanza begins with a line containing the stanza name, which is the name of a resource class, the keyword **OTHER**, or the keyword **DEFAULT**.

The **OTHER** stanza applies to all resource classes that are not otherwise specified in the file. The lines in the **DEFAULT** stanza are implicitly appended to the stanzas for each resource class specified in the ACL file, including the **OTHER** stanza. If the **OTHER** stanza is not specified, then the permissions of any resource class not specified in this file will be the permissions specified in the **DEFAULT** stanza.

The line containing the stanza name must start in column one. The remaining lines of the stanza, excluding comment lines, consists of leading white space (one or more blanks, tabs, or both) followed by one or more white-space separated tokens that include:

- a user identifier,
- an object type,
- and an optional set of permissions.

The `user_identifier` portion of the stanza line can have any one of the forms shown in the following table:

Table 18. The user identifier portion of the stanza line

This Form:	Specifies:								
host: <i>host_user_identifier</i>	<p>A host user identifier. The host: keyword is optional. It specifies that the user identifier can be matched against a network identifier provided by the Host Based Authentication (HBA) security mechanism (described in Chapter 6, “Understanding and administering cluster security services,” on page 147). If the host: keyword is omitted and the entry does not take one of the other forms outlined in this table, the entry is assumed to be a host user identifier.</p> <p>The host user identifier can take a number of different forms.</p>								
	<table border="1"> <thead> <tr> <th>This host user identifier format:</th> <th>Specifies:</th> </tr> </thead> <tbody> <tr> <td><i>user_name@host_identifier</i></td> <td>A particular user. The <i>host_identifier</i> portion of this specification can take a number of forms. These forms are the same as when the host user identifier format is specified as a <i>host_identifier</i> alone, and are described below.</td> </tr> <tr> <td><i>host_identifier</i></td> <td> <p>Any user running the RMC application on the host identified. The <i>host_identifier</i> can be:</p> <ul style="list-style-type: none"> • a fully qualified host name • a short host name • an IP address • a node ID. This is a 16-digit hexadecimal number. For example, 0xaf58d41372c47686. • the keyword LOCALHOST. This keyword identifies the local host. </td> </tr> <tr> <td>*</td> <td>Any user running an RMC application on any host.</td> </tr> </tbody> </table>	This host user identifier format:	Specifies:	<i>user_name@host_identifier</i>	A particular user. The <i>host_identifier</i> portion of this specification can take a number of forms. These forms are the same as when the host user identifier format is specified as a <i>host_identifier</i> alone, and are described below.	<i>host_identifier</i>	<p>Any user running the RMC application on the host identified. The <i>host_identifier</i> can be:</p> <ul style="list-style-type: none"> • a fully qualified host name • a short host name • an IP address • a node ID. This is a 16-digit hexadecimal number. For example, 0xaf58d41372c47686. • the keyword LOCALHOST. This keyword identifies the local host. 	*	Any user running an RMC application on any host.
This host user identifier format:	Specifies:								
<i>user_name@host_identifier</i>	A particular user. The <i>host_identifier</i> portion of this specification can take a number of forms. These forms are the same as when the host user identifier format is specified as a <i>host_identifier</i> alone, and are described below.								
<i>host_identifier</i>	<p>Any user running the RMC application on the host identified. The <i>host_identifier</i> can be:</p> <ul style="list-style-type: none"> • a fully qualified host name • a short host name • an IP address • a node ID. This is a 16-digit hexadecimal number. For example, 0xaf58d41372c47686. • the keyword LOCALHOST. This keyword identifies the local host. 								
*	Any user running an RMC application on any host.								
none: <i>mapped_user_identifier</i>	A mapped name as specified in the ctsec_map.global or ctsec_map.local file. See “Configuring the Host Based Authentication mechanism mappings” on page 163 for more information on creating these mapped names.								
UNAUTHENT	An unauthenticated user.								

The stanza, including lines implicitly appended from the DEFAULT stanza, is examined in two passes. The first pass attempts to match a line against the user’s network ID. If no match can be made, then a second pass is performed in an attempt to match a line against the user’s mapped ID.

The next part of the stanza is the type; it can be any of the characters shown in the following table.

Table 19. The type portion of the stanza line

Specifying this:	Indicates that the permissions provide access to:
C	the resource class
R	all resource instances of the class
*	both the resource class and all instances of the class

The final part of the stanza line is the optional permissions.

Table 20. The optional permissions portion of the stanza line

Specifying this:	Indicates that the specified user(s) at the specified host(s) have:
r	read permission. This allows the user(s) to register and unregister events, query attribute values, and validate resource handles.

Table 20. The optional permissions portion of the stanza line (continued)

Specifying this:	Indicates that the specified user(s) at the specified host(s) have:
w	write permission. This allows the user(s) to run all other command interfaces.
rw	read and write permission.

If the permissions are omitted, then the user does not have access to the objects specified by the *type* character. 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 order specified in the ACL file. The first line that contains an identifier that matches the user issuing the command and an object type that matches the objects specified by the command is the line used in determining access permissions. Therefore, lines containing more specific user identifiers and object types should be placed before lines containing less specific user identifiers and object types.

How to modify the ACL file

When RMC is installed on a node, a default ACL file is provided in **/usr/sbin/rsct/cfg/ctrmc.acls**. This file **should not be modified**. It contains the following default permissions.

```
IBM.HostPublic
    *          *      r
    UNAUTHENT *      r

DEFAULT
    root@LOCALHOST * rw
    LOCALHOST * r
```

The first stanza enables anyone to read the information in the IBM.HostPublic class which provides information about the node, mainly its public key. The second stanza contains default ACL entries. It grants, for this node, read/write permission to root and read-only permission to any other user.

To change these defaults:

1. Copy the **/usr/sbin/rsct/cfg/ctrmc.acls** file to **/var/ct/cfg/ctrmc.acls**

```
cp /usr/sbin/rsct/cfg/ctrmc.acls /var/ct/cfg/ctrmc.acls
```
2. Using an ASCII text editor, modify the new **ctrmc.acls** file in **/var/ct/cfg/**. Refer to “Format of an ACL file” on page 77 for information on how to construct the file stanzas.
3. Activate your new permissions using the **refresh** command.

```
refresh -s ctrmc
```

Provided there are no errors in the modified ACL file, the new permissions will take effect. If errors are found in the modified ACL file, then the contents of the file are ignored and the previously-defined permissions remain in effect. The ACL file errors are logged to **/var/ct/IW/log/mc/default**.

Listing Resource Information

The **lsrsrc** and **lsrsrcdef** commands enable you to list information about the resources available on your system or cluster. Specifically, you can:

- issue either the **lsrsrc** or **lsrsrcdef** command without any command parameters or flags to obtain a list of resource classes available on your system or cluster. See “Listing available resource classes” for more information.
- use the **lsrsrc** command to list the values of resource or resource class attributes. See “Displaying attribute value information for a resource or a resource class” on page 81 for more information.
- use the **lsrsrcdef** command to list attribute definitions for a resource or resource class. See “Displaying attribute definition information for a resource or a resource class” on page 83 for more information.

Listing available resource classes

To display a list of the resource classes on your cluster or system, issue either the **lsrsrc** or **lsrsrcdef** command without any command parameters or flags.

```
lsrsrc
```

Output will be similar to the following:

```
class_name
"IBM.Association"
"IBM.ATMDevice"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EthernetDevice"
"IBM.EventResponse"
"IBM.FDDIDevice"
"IBM.Host"
"IBM.FileSystem"
"IBM.PagingDevice"
"IBM.PhysicalVolume"
"IBM.Processor"
"IBM.Program"
"IBM.TokenRingDevice"
"IBM.Sensor"
"IBM.PeerDomain"
"IBM.PeerNode"
"IBM.RSCTParameters"
"IBM.NetworkInterface"
"IBM.CommunicationGroup"
"IBM.HostPublic"
"IBM.TieBreaker"
"IBM.LPCommands"
```

To return detailed information on any of the resource classes, refer to the instructions in “Displaying attribute value information for a resource or a resource class” on page 81 and “Displaying attribute definition information for a resource or a resource class” on page 83.

The following table lists the resource classes defined by RSCT resource managers. In addition to the resource classes listed in the table, keep in mind that:

- any resource classes registered using the CIM resource manager (described in “Querying CIM properties” on page 122) may also appear in the preceding command output. When a CIM class is registered, it is mapped to a new RMC resource class. The RMC resource class name will be a concatenation of the namespace and the CIM class name — for example *cimv2.Linux_ComputerSystem*. All registered CIM classes are placed in the root/cimv2 namespace.
- additional resource managers are provided by cluster licensed program products that exploit the RSCT technology (such as CSM, which provides the Domain

Management resource manager). If you have any RSCT exploiter products installed on your system or cluster, the preceding command output may show additional resource classes that are defined by resource managers of the RSCT exploiter(s). For information on any additional resource classes not described in the following table, refer to the appropriate RSCT exploiter documentation.

Table 21. Resource classes provided by RSCT resource managers

This resource class	Is defined by this resource manager	For more information, see:
IBM.Association	Event Response resource manager	Table 11 on page 68
IBM.ATMDevice	Host resource manager	Table 13 on page 69
IBM.AuditLog	Audit Log resource manager	Table 9 on page 66
IBM.AuditLogTemplate	Audit Log resource manager	Table 9 on page 66
IBM.CommunicationGroup	Configuration resource manager	Table 10 on page 67
IBM.Condition	Event Response resource manager	Table 11 on page 68
IBM.EthernetDevice	Host resource manager	Table 13 on page 69
IBM.EventResponse	Event Response resource manager	Table 11 on page 68
IBM.FDDIDevice	Host resource manager	Table 13 on page 69
IBM.FileSystem	File System resource manager	Table 12 on page 68
IBM.Host	Host resource manager	Table 13 on page 69
IBM.HostPublic	Host resource manager	Table 13 on page 69
IBM.LPCCommands	Least-privilege resource manager	Table 14 on page 69
IBM.NetworkInterface	Configuration resource manager	Table 10 on page 67
IBM.PagingDevice	Host resource manager	Table 13 on page 69
IBM.PeerDomain	Configuration resource manager	Table 10 on page 67
IBM.PeerNode	Configuration resource manager	Table 10 on page 67
IBM.PhysicalVolume	Host resource manager	Table 13 on page 69
IBM.Processor	Host resource manager	Table 13 on page 69
IBM.Program	Host resource manager	Table 13 on page 69
IBM.RSCTParameters	Configuration resource manager	Table 10 on page 67
IBM.Sensor	Sensor resource manager	Table 15 on page 69
IBM.TieBreaker	Configuration resource manager	Table 10 on page 67
IBM.TokenRingDevice	Host resource manager	Table 13 on page 69

For complete syntax information on the **lsrsrc** and **lsrsrcdef** commands, refer their online man pages. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Displaying attribute value information for a resource or a resource class

You can display attribute value information for a resource or a resource class by issuing the **lsrsrc** command with the name of a resource class. Various command options, as shown in the following table, enable you to display attribute information

- for the resource class or resource instances of the class
- for persistent attributes, dynamic attributes, or both persistent and dynamic attributes

Table 22. Displaying attribute value information

To display attribute value information...	for persistent attributes:	for dynamic attributes:	for both persistent and dynamic attributes:
for a resource class:	<code>lsrsrc -c -A p Resource_Class</code>	<code>lsrsrc -c -A d Resource_Class</code>	<code>lsrsrc -c -A b Resource_Class</code>
for a resource:	<code>lsrsrc -A p Resource_Class</code>	<code>lsrsrc -A d Resource_Class</code>	<code>lsrsrc -A b Resource_Class</code>

The **lsrsrc** command will return a list of the attributes requested. The attribute name and value will be listed. For example, to list the persistent and dynamic attributes of for resources of the IBM.Host class, you would enter:

```
lsrsrc -A b IBM.Host
```

Output would be similar to the following:

```
Resource Persistent and Dynamic Attributes for IBM.Host
resource 1:
```

```

Name           = "jbrady.ibm.com"
NumProcessors  = 4
RealMemSize    = 1073676288
OSName         = "AIX"
KernelVersion  = "5.3"
DistributionName = "IBM"
DistributionVersion = "5.3"
Architecture   = "ppc"
NumOnlineProcessors = 4
ActivePeerDomain = "JoeMixed"
NodeNameList   = {"jbrady.ibm.com"}
ProcRunQueue   = 1.01167
ProcSwapQueue  = 1.01822
TotalPgSpSize  = 131072
TotalPgSpFree  = 130845
PctTotalPgSpUsed = 0.173187
PctTotalPgSpFree = 99.8268
PctTotalTimeIdle = 95.0711
PctTotalTimeWait = 0.152439
PctTotalTimeUser = 4.06504
PctTotalTimeKernel = 0.711382
PctRealMemFree  = 58
PctRealMemPinned = 8
RealMemFramesFree = 153110
```

```

.
.
.
```

Although the attribute names are often self-explanatory, you can use the **lsrsrcdef** command to display definition information (including a description) for the attributes listed. The **lsrsrcdef** command is described in “Displaying attribute definition information for a resource or a resource class” on page 83.

Targeting Node(s):

The **lsrsrc** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope. The **lsrsrc** command’s **-a** flag, if specified, indicates that the command applies to all nodes in the management scope. If the **CT_MANAGEMENT_SCOPE** environment variable is not set and the **-a** flag is specified, then the default management scope will be the management

domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **lsrsrc** command man page and “How do I determine the target nodes for a command?” on page 75.

For complete syntax information on the **lsrsrc** command, refer its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Displaying attribute definition information for a resource or a resource class

You can display attribute definition information for a resource or a resource class by issuing the **lsrsrcdef** command with the name of a resource class. Various command options, as shown in the following table, enable you to display attribute definition information

- for the resource class or resource instances of the class
- for persistent attributes or dynamic attributes

Table 23. Displaying attribute definition information

To display attribute definition information...	for persistent attributes:	for dynamic attributes:
for a resource class:	<code>lsrsrcdef -c -A p -p 0 -e Resource_Class</code>	<code>lsrsrcdef -c -A d -p 0 -e Resource_Class</code>
for a resource:	<code>lsrsrcdef -A p -p 0 -e Resource_Class</code>	<code>lsrsrcdef -A d -p 0 -e Resource_Class</code>

The **lsrsrcdef** commands shown in the preceding table will return the definition for each persistent or dynamic attribute of the requested resource class or resource. The following example shows the attribute definition returned for two attributes of the IBM.Host resource. The **-e** flag in the preceding table’s commands specifies expanded output format. The expanded format includes the description field shown in the following command output.

```
attribute 7:
    program_name = "OSName"
    display_name = "Operating System Name"
    group_name   = "General"
    properties   = {"read_only","inval_for_define","selectable","public"}
    description  = "This attribute reflects the name of the operating syste
m running on the node (e.g. Linux, AIX, ...)."
```

```
attribute 8:
    program_name = "KernelVersion"
    display_name = "Kernel Version"
    group_name   = "General"
    properties   = {"read_only","inval_for_define","selectable","public"}
    description  = "This attribute reflects the version of the operating sy
stem kernel running
on the node."
```

```
attribute_id = 6
group_id     = 0
data_type    = "char_ptr"
variety_list = {[1,5]}
variety_count = 1
default_value = ""
```

```
attribute_id = 7
group_id     = 0
```

```
data_type      = "char_ptr"
variety_list   = {[1,5]}
variety_count  = 1
default_value  = ""
```

If you want to return the definition of specific attributes only, simply include the attribute name(s) on the **lsrsrcdef** command line. For example:

```
lsrsrcdef -e IBM.Host KernelVersion
```

Targeting Node(s):

The **lsrsrcdef** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope. The **lsrsrcdef** command's **-a** flag, if specified, indicates that the command applies to all nodes in the management scope. If the **CT_MANAGEMENT_SCOPE** environment variable is not set and the **-a** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **lsrsrcdef** command man page and “How do I determine the target nodes for a command?” on page 75.

For complete syntax information on the **lsrsrcdef** command, refer its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Basic resource monitoring

This section describes the Event Response Resource Manager commands you can use for monitoring your system of cluster domain. As described in “How do RMC and the resource managers enable you to monitor resources?” on page 70, you can monitor events of interest (called *conditions*) and have the RMC system react in particular ways (called *responses*) if the event occurs. To do this you create a condition/response association using the **mkcondresp** command, and then issue the **startcondresp** command to start monitoring the condition. Using the **CT_MANAGEMENT_SCOPE** environment variable, you can determine the set of nodes that will be monitored — either the local node only, the nodes in a peer domain, or the nodes in a management domain.

This section is called “Basic Monitoring” because it covers monitoring using only predefined conditions and responses. It describes how to:

- List conditions, responses, and condition/response associations using the **lscondition**, **lsresponse**, and **lscondresp** commands.
- Create a condition/response association using the **mkcondresp** command.
- Start condition monitoring using the **startcondresp** command.
- Stopping condition monitoring using the **stopcondresp** command.
- Removing a condition/response association using the **rmcondresp** command.

For information on creating your own conditions and responses rather than using the predefined ones provided by the various resource managers, refer to “Advanced resource monitoring” on page 98. For detailed syntax information on any the

commands described in this section, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Listing conditions, responses, and condition/response associations

There are three commands for listing condition and response information. These are useful when working with conditions, responses, and condition/response associations. These commands are:

- **Iscondition** for listing information about conditions.
- **Isresponse** for listing information about responses.
- **Iscondresp** for listing information about condition/response associations.

Listing conditions

For a list of all available conditions, enter the **Iscondition** command. For example, entering the following at the command prompt:

```
Iscondition
```

Results in output similar to the following:

Name	MonitorStatus
"FileSystem space used"	"Not monitored"
"tmp space used"	"Not monitored"
"var space used"	"Not monitored"

Results will differ depending on what resource managers are available. The list will include any predefined conditions provided by the various resource managers, and also any conditions you create (as described in “Creating a condition” on page 101). The "MonitorStatus" in the preceding output indicates whether or not the condition is currently being monitored.

To list more detailed information about a particular condition, specify its name as a parameter to the **Iscondition** command. For example, to get detailed information about the "FileSystem space used" condition, enter the following at the command prompt:

```
Iscondition "FileSystem space used"
```

Results will be similar to the following:

```
Name = "FileSystem space used"
Location = "nodeA"
MonitorStatus = "Monitored"
ResourceClass = "IBM.FileSystem"
EventExpression = "PercentTotUsed > 99"
EventDescription = "Generate event when space used is
greater than 99 percent full"
RearmExpression = "PercentTotUsed < 85"
RearmDescription = "Start monitoring again after it is
less than 85 percent"
SelectionString = ""
Severity = "w"
NodeNameList = {}
MgtScope = "1"
```

Targeting Node(s):

The **Iscondition** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope

— either local scope, peer domain scope, or management domain scope. The **Iscondition** command's **-a** flag, if specified, indicates that the command applies to all nodes in the management scope. If the `CT_MANAGEMENT_SCOPE` environment variable is not set and the **-a** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **Iscondition** command man page and "How do I determine the target nodes for a command?" on page 75.

For detailed syntax information on the **Iscondition** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Listing responses

For a list of all available responses, enter the **Isresponse** command. For example, entering the following at the command prompt:

```
Isresponse
```

Results in output similar to the following:

```
Name
"E-mail root any time"
"E-mail root first shift"
"Critical notifications"
"Generate SNMP trap"
```

Results will differ depending on what resource managers are available. The list will include any predefined responses provided by the various resource managers, and also any responses you create (as described in "Creating a response" on page 114).

To list more detailed information about a particular response, specify its name as a parameter to the **Isresponse** command. For example, to get detailed information about the "Informational notifications" response, enter the following at the command prompt:

```
Isresponse "Informational notifications"
```

This displays the following output showing details for the two actions associated with this response.

Displaying response information:

```
ResponseName = "Informational notifications"
Node         = "c175n06.ppd.pok.ibm.com"
Action       = "Log info event"
DaysOfWeek   = 1-7
TimeOfDay    = 0000-2400
ActionScript = "/usr/sbin/rsct/bin/logevent /tmp/infoEvents"
ReturnCode   = -1
CheckReturnCode = "n"
EventType    = "b"
StandardOut  = "n"
EnvironmentVars = ""
UndefRes     = "n"

ResponseName = "Informational notifications"
Node         = "c175n06.ppd.pok.ibm.com"
Action       = "E-mail root"
```

```

DaysOfWeek      = 2-6
TimeOfDay       = 0800-1700
ActionScript    = "/usr/sbin/rsct/bin/notifyscript root"
ReturnCode      = -1
CheckReturnCode = "n"
EventType       = "b"
StandardOut     = "n"
EnvironmentVars = ""
UndefRes        = "n"

```

Targeting Node(s):

The **lsresponse** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope. The **lsresponse** command's **-a** flag, if specified, indicates that the command applies to all nodes in the management scope. If the **CT_MANAGEMENT_SCOPE** environment variable is not set and the **-a** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **lsresponse** command man page and “How do I determine the target nodes for a command?” on page 75.

For detailed syntax information on the **lsresponse** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Listing condition/response associations

As described in “Listing conditions” on page 85 and “Listing responses” on page 86, many predefined conditions and responses are provided by the various resource managers on your system. What’s more, you can create your own conditions and responses as described in “Advanced resource monitoring” on page 98. Before you can monitor a condition, however, you must link it with one or more responses. This is called a condition/response association, and is required for monitoring so that RMC knows how to respond when the condition event occurs.

For a list of all available condition/response associations, enter the **lscondresp** command. For example, if no condition/response associations have been created, entering the following at the command prompt:

```
lscondresp
```

Results in the output:

```
lscondresp: No defined condition-response links were found
```

Once you link conditions with responses (as described in “Creating a condition/response association” on page 88), entering the **lscondresp** command will show the associations. For example:

Condition	Response	State	Location
"FileSystem space used"	"Broadcast event on-shift"	"Active"	nodeA
"FileSystem space used"	"E-mail root any time"	"Not Active"	nodeA
"Page in Rate"	"Log event any time"	"Active"	nodeA

If you want to list the condition/response associations for a single condition, supply the condition name as a parameter to the **Iscondresp** command. For example, to list the condition/response associations for the "FileSystem space used" condition, you would enter the following at the command prompt:

```
Iscondresp "FileSystem space used"
```

Output would be similar to the following:

Condition	Response	State	Location
"FileSystem space used"	"Broadcast event on-shift"	"Active"	nodeA
"FileSystem space used"	"E-mail root any time"	"Not Active"	nodeA

If you wanted to limit the preceding output to show just the active condition/response associations, you would use the **Iscondresp** command's **-a** option. For example:

```
Iscondresp -a "FileSystem space used"
```

Output would show only the active condition/response associations for the "FileSystem space used" condition.

Condition	Response	State	Location
"FileSystem space used"	"Broadcast event on-shift"	"Active"	nodeA

Targeting Node(s):

The **Iscondresp** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope. The **Iscondresp** command's **-z** flag, if specified, indicates that the command applies to all nodes in the management scope. If the **CT_MANAGEMENT_SCOPE** environment variable is not set and the **-z** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **Iscondresp** command man page and "How do I determine the target nodes for a command?" on page 75.

For detailed syntax information on the **Iscondresp** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating a condition/response association

Before you can monitor a condition, you must link it with one or more responses. This is called a condition/response association, and is required for monitoring so that RMC knows how to respond when the condition event occurs. Many predefined conditions and responses are provided by the various resource managers on your system. What's more, you can create your own conditions and responses as described in "Advanced resource monitoring" on page 98. To list all the available conditions you can use in creating your condition/response association, use the **Iscondition** command as described in "Listing conditions" on page 85. To list all the available responses you can use in creating your condition/response association, use the **Isresponse** command as described in "Listing responses" on page 86.

To create a condition/response association, use the **mkcondresp** command. The **mkcondresp** command links responses with a condition, but does not start monitoring of the condition. To create the condition/response association and start monitoring the condition, use the **startcondresp** command (described next in “Starting condition monitoring” on page 90).

To use the **mkcondresp** command to link the condition "FileSystem space used" with the response "Broadcast event on-shift", enter the following at the command prompt:

```
mkcondresp "FileSystem space used" "Broadcast event on-shift"
```

You can also specify multiple responses that you want to associate with the condition. For example, the following example links both the "Broadcast event on-shift" and "E-mail root any time" responses with the "FileSystem space used" condition.

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

When monitoring in a management domain or peer domain scope, the condition and response you link must be defined on the same node. By default, the **mkcondresp** command assumes this is the local node. If they are defined on another node, you can specify the node name along with the condition. For example:

```
mkcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
```

Although you specify the node name on the condition, be aware that both the condition and response must be defined on that node.

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the CT_MANAGEMENT_SCOPE environment variable) for the local node or the node specified by the CT_CONTACT environment variable (if it is set). For more information, refer to the **mkcondresp** command man page and “How do I determine the target nodes for a command?” on page 75.

Once you have linked one or more responses with a condition using the **mkcondresp**, you can verify that the condition/response association has been created by issuing the **lscondresp** command (as described in “Listing condition/response associations” on page 87).

The **mkcondresp** command links responses with a condition, but does not start monitoring of the condition. To start monitoring the condition, use the **startcondresp** command (described next in “Starting condition monitoring” on page 90).

To prevent user modification or removal of a condition/response link, you can lock it (as described in “Locking and unlocking conditions, responses, and condition/response links” on page 128).

For detailed syntax information on the **mkcondresp** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Starting condition monitoring

The **startcondresp** command starts monitoring a condition that has one or more linked responses. If you have already created these condition/response associations using the **mkcondresp** command (as described in “Creating a condition/response association” on page 88), you can simply specify the name of the condition you want to start monitoring as a parameter of the **startcondresp** command. For example, entering the following at the command prompt:

```
startcondresp "FileSystem space used"
```

Starts monitoring the condition "FileSystem space used" using all of its linked responses.

For a list of all the available condition/response associations already defined, you can issue the **lscondresp** command as described in “Listing condition/response associations” on page 87. The listing returned by the **lscondresp** command also shows the state of the condition/response association (active or not active), so you can use it to verify that monitoring has started.

If a condition has multiple linked responses, and you do not want RMC to use all of them, you can explicitly state which response you want triggered when the condition is true. You do this by specifying the responses as parameters to the **startcondresp** command. For example, if the "FileSystem space used" condition has multiple responses linked with it, you could start monitoring that will use just the "Broadcast event on-shift" response by entering the following at the command prompt:

```
startcondresp "FileSystem space used" "Broadcast event on-shift"
```

If you wanted to also use the "E-mail root any time" response, you would enter:

```
startcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root any time"
```

You can also use the above format of specifying a response on the **startcondresp** command to create a condition/response association and start monitoring in one step. If the "FileSystem space used" condition had not already been linked with the "Broadcast event on-shift" response, then the command:

```
startcondresp "FileSystem space used" "Broadcast event on-shift"
```

would create the association and start monitoring. In this way, the **startcondresp** command is like the **mkcondresp** command. The difference is that the **mkcondresp** command merely creates the condition/response association, while the **startcondresp** command creates the association and starts monitoring in one step.

If using the **startcondresp** command to create a command/response association, be aware that, when monitoring in a management domain or peer domain scope, the condition and response you link must be defined on the same node. By default, the **startcondresp** command assumes this is the local node. If they are defined on another node, you can specify the node name along with the condition. For example:

```
startcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
```

Although you specify the node name on the condition, but be aware that both the condition and response must be defined on that node.

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the CT_MANAGEMENT_SCOPE environment variable) for the local node or the node specified by the CT_CONTACT environment variable (if it is set). For more information, refer to the **startcondresp** command man page and “How do I determine the target nodes for a command?” on page 75.

To prevent a user from stopping monitoring, you can lock the condition/response link (as described in “Locking and unlocking conditions, responses, and condition/response links” on page 128). Locking a condition/response link also prevents accidental removal of the link.

For detailed syntax information on the **startcondresp** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Stopping condition monitoring

The **stopcondresp** command stops monitoring of a condition that has one or more linked responses.

For example, to stop all active responses for the "FileSystem space used" condition, you would enter the following at the command prompt:

```
stopcondresp "FileSystem space used"
```

If you are unsure which conditions are currently being monitored, you can use the **lscndition** command as described in “Listing conditions” on page 85.

If the condition has multiple linked and active responses, and you only want to stop a selection of those responses, while allowing the other responses to remain active, simply specify the response(s) you want to deactivate as parameters on the **stopcondresp** command. (To ascertain which responses are active for the condition, use the **lscndresp** command as described in “Listing condition/response associations” on page 87.) If you wanted to deactivate the "Broadcast event on-shift" response for the "FileSystem space used" condition, you would enter the following at the command prompt:

```
stopcondresp "FileSystem space used" "Broadcast event on-shift"
```

If you wanted to deactivate the responses "Broadcast event on-shift" and "E-mail root any time" for the "FileSystem space used" condition, you would enter:

```
stopcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root any time"
```

If the condition/response link you specify on the **stopcondresp** command is locked, monitoring will not be stopped; instead, an error will be generated informing you that the condition/response link is locked. For information on unlocking a condition/response link so monitoring can be stopped, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

Targeting Node(s):

If the condition you want to stop monitoring is defined on another node, you can specify the node name along with the condition. For example:

```
stopcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
```

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the `CT_MANAGEMENT_SCOPE` environment variable) for the local node or the node specified by the `CT_CONTACT` environment variable (if it is set). For more information, refer to the **stopcondresp** command man page and “How do I determine the target nodes for a command?” on page 75.

For detailed syntax information on the **stopcondresp** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing a condition/response association

The **rmcondresp** command enables you to remove a condition/response association. To see a list of the existing condition/response associations that you can remove, you can use the **lscondresp** command as described in “Listing condition/response associations” on page 87. The **rmcondresp** command enables you to remove a specified condition/response association, all the associations for a specified condition, or all the associations for a specified response.

To remove a specific condition/response association, specify both the condition and response as parameters to the **rmcondresp** command. For example, the following command deletes the link between the "FileSystem space used" condition and the "Broadcast event on-shift" response.

```
rmcondresp "FileSystem space used" "Broadcast event on-shift"
```

You can also delete the links between a condition and multiple responses. For example, the following command deletes the links between the "FileSystem space used" condition and the responses "Broadcast event on-shift" and "E-mail root any time":

```
rmcondresp "FileSystem space used" "Broadcast event on-shift" "E-mail root any time"
```

To remove links to all responses associated with a particular condition, specify the condition only as a parameter to the **rmcondresp** command. For example, to remove the links to all responses associated with the "FileSystem space used" condition, you would enter the following at the command prompt:

```
rmcondresp "FileSystem space used"
```

Similarly, you can remove all links to one or more responses using the **rmcondresp** command's **-r** option. The **-r** option tells the **rmcondresp** command that all the command parameters are responses. In the following command example, all links to the "Broadcast event on-shift" response are removed:

```
rmcondresp -r "Broadcast event on-shift"
```

You can also specify multiple responses. The following example removes all condition/response associations that use the "Broadcast event on-shift" or "E-mail root any time" responses.

```
rmcondresp -r "Broadcast event on-shift" "E-mail root any time"
```

If the condition/response link you specify on the **rmcondresp** command is locked, it will not be removed; instead, an error will be generated informing you that the condition/response link is locked. For information on unlocking the condition/response link so it can be removed, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

Targeting Node(s):

If the condition and response you want to stop monitoring are defined on another node, you can specify the node name along with the condition. For example:

```
rmcondresp "FileSystem space used":nodeA "Broadcast event on-shift"
```

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the `CT_MANAGEMENT_SCOPE` environment variable) for the local node or the node specified by the `CT_CONTACT` environment variable (if it is set). For more information, refer to the **rmcondresp** command man page and “How do I determine the target nodes for a command?” on page 75.

For detailed syntax information on the **rmcondresp** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Using the audit log to track monitoring activity

When you are monitoring a condition, you should be aware that any linked response actions will be executed in the background by daemons. Often, the response action will somehow log or notify you about the event occurring. For example, all of the predefined responses, use response scripts we provide that either:

- logs information to a file,
- mails the information to a particular user ID, or
- broadcasts the information to all users who are logged in.

In some cases, you might create your own response script that performs no such logging or notification, but instead provides a more targeted solution for the monitored attribute testing true. For example, you might create a recovery script that deletes unnecessary files when the **/tmp** directory is 90% full.

Whether or not the response script performs some type of notification or logging itself, it is important to know that RMC has an audit log that it uses to record information about the system’s operation, and that the Event Response Resource Manager appends entries for all triggered response actions to this log. The audit log includes information about the normal operation of the system as well as failures and other errors, and so augments any information that a response script might provide.

To list records from the audit log, use the **lsaudrec** command. For example, to list all records in the audit log, enter:

```
lsaudrec
```

Output will be similar to the following:

```
Time                Subsystem Category Description
07/27/02 14:55:42    ERRM Info    Monitoring of condition Processor idle time
                is started successfully.
07/27/02 14:55:58    ERRM Info    Event : Processor idle time occurred at 07/
27/02 14:55:58 953165 on proc0 on c175n06.ppd.pok.ibm.com.
07/27/02 14:55:59    ERRM Info    Event from Processor idle time that occurre
d at 07/27/02 14:55:58 953165 will cause /usr/sbin/rsct/bin/logevent /tmp/system
Events from Log event anytime to be executed.
07/27/02 14:55:59    ERRM Info    Event : Processor idle time occurred at 07/
```

```

27/02 14:55:58 953165 on proc1 on c175n06.ppd.pok.ibm.com.
07/27/02 14:55:59      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 will cause /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to be executed.
07/27/02 14:55:59      ERRM Info      Event : Processor idle time occurred at 07/27/02 14:55:58 953165 on proc2 on c175n06.ppd.pok.ibm.com.
07/27/02 14:55:59      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 will cause /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to be executed.
07/27/02 14:55:59      ERRM Info      Event : Processor idle time occurred at 07/27/02 14:55:58 953165 on proc3 on c175n06.ppd.pok.ibm.com.
07/27/02 14:55:59      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 will cause /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to be executed.
07/27/02 14:56:00      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to complete with a return code of 0.
07/27/02 14:56:00      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to complete with a return code of 0.
07/27/02 14:56:00      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to complete with a return code of 0.
07/27/02 14:56:00      ERRM Info      Event from Processor idle time that occurred at 07/27/02 14:55:58 953165 caused /usr/sbin/rsct/bin/logevent /tmp/systemEvents from Log event anytime to complete with a return code of 0.
07/27/02 14:56:51      ERRM Info      Monitoring of condition Processor idle time is stopped successfully.

```

The above example shows:

- when RMC started monitoring the "Processor idle time" condition
- each time the "Processor idle time" condition tested true
- that the "Log event anytime" response was associated with the "Processor idle time" condition, and as a result, its response action `"/usr/sbin/rsct/bin/logevent /tmp/systemEvents"` was executed each time the "Processor idle time" condition tested true.
- The return code from each execution of the command `"/usr/sbin/rsct/bin/logevent /tmp/systemEvents"`
- when RMC stopped monitoring the "Processor idle time" condition.

The above audit log is quite small and contains entries related to a single monitored condition. In practice, however, the audit log is likely to contain a very large number of records. For this reason, the **lsaudrec** command enables you to filter the audit log so that only a subset of its records are returned.

To filter the audit log, use the **lsaudrec** command's **-s** option followed by a *selection string* — an expression that determines how the audit log is to be filtered. Every record in the audit log has a number of named fields (such as **Time**) that provide specific information associated with the record. These field names are used in the selection string expression, which also includes constants and operators. Expressions in RMC are discussed in more detail in "Using expressions to specify condition events and command selection strings" on page 130. Here it suffices to say that the syntax of the selection string is similar to an expression in the C programming language or the *where* clause in SQL. The selection string you provide is matched against each record in the audit log. The **lsaudrec** man page contains detailed syntax information on the **-s** option and the field names you can use when filtering the audit log. Here we will discuss only the most common field names you would typically use when filtering the audit log.

For example, you would commonly want to filter the audit log based on the time records were created. You can do this using the **-s** flag and the **Time** field name. To filter the audit log so that only records created on July 27 between 14:30 and 15:00 are listed, you would enter the following command:

```
lsaudrec -s "Time > #072714302002 && Time < #072715002002"
```

The expression used in the preceding example specifies the date/time using the format `#mddhhmmyyyy`, where, from left to right: `mm` = month, `dd` = day, `hh` = hour, `mm` = minutes, and `yyyy` = year. The fields can be omitted from right to left. If not present, the following defaults are used: year = the current year, minutes = 00, hour = 00, day = 01, and month = the current month. This next example omits the year information:

```
lsaudrec -s "Time > #07271430 && Time < #07271500"
```

You can also specify the time using the format `#-mddhhmmyyy`. In this case, the time specified is relative to the current time. Again, fields can be omitted from right to left; for this format the omitted fields are replaced by 0. So, for example, the value `#-0001` corresponds to one day ago, and the value `#-010001` corresponds to one month and one hour ago. To list the audit log entries that were logged in the last hour only, you would enter:

```
lsaudrec -s "Time > #-000001"
```

Another field that is commonly used when filtering the audit log is the **Category** field. If the **Category** field of an audit log record is 0, it is an informational message. If the **Category** field of an audit log record is 1, it is an error message. To list just the error messages in an audit log, you would enter:

```
lsaudrec -s "Category=1"
```

Targeting Node(s):

The **lsaudrec** command is affected by the environment variables `CT_CONTACT` and `CT_MANAGEMENT_SCOPE`. The `CT_CONTACT` environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The `CT_MANAGEMENT_SCOPE` indicates the management scope — either local scope, peer domain scope, or management domain scope.

The **lsaudrec** command's **-a** flag, if specified, indicates that the command applies to all nodes in the management scope.

The **lsaudrec** command's **-n** flag specifies a list of nodes containing the audit log records to display. Any node specified must be within the management scope (as determined by the `CT_MANAGEMENT_SCOPE` environment variable) for the local node or the node specified by the `CT_CONTACT` environment variable (if it is set).

If the `CT_MANAGEMENT_SCOPE` environment variable is not set and either the **-a** flag or **-n** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **lsaudrec** command man page and "How do I determine the target nodes for a command?" on page 75.

For detailed syntax information on the **lsaudrec** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Deleting entries from the audit log

There are two ways to delete entries from the audit log — explicitly (using the **rmaudrec** command) or implicitly (by setting the **RetentionPeriod** and **MaxSize** attributes of the IBM.AuditLog resource).

Deleting entries from the audit log using the rmaudrec command: The **rmaudrec** command removes records from the audit log. You must provide this command with a *selection string* — an expression that indicates which records should be deleted. Like the **Isaudrec** command, the **rmaudrec** command has an **-s** option for specifying the selection string expression, which takes the same form as it does on the **Isaudrec** command. For example, to remove all records from the audit log, you would enter:

```
rmaudrec -s "Time > 0"
```

To remove only the records that were created on July 27 between 14:30 and 15:00, you would enter:

```
rmaudrec -s "Time > #07271430 && Time < #07271500"
```

To delete the audit log entries that were logged in the last hour only, you would enter:

```
rmaudrec -s "Time > #-000001"
```

To remove only informational messages from the audit log (leaving error messages), you would enter:

```
rmaudrec -s "Category=0"
```

Targeting Node(s):

The **rmaudrec** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope.

The **rmaudrec** command's **-a** flag, if specified, indicates that the command applies to all nodes in the management scope.

The **rmaudrec** command's **-n** flag specifies a list of nodes whose audit log records can be deleted (if they meet other criteria such as matching the selection string). Any node specified must be defined within the management scope (as determined by the **CT_MANAGEMENT_SCOPE** environment variable) for the local node or the node specified by the **CT_CONTACT** environment variable (if it is set).

If the **CT_MANAGEMENT_SCOPE** environment variable is not set and either the **-a** flag or **-n** flag is specified, then the default management scope will be the management domain scope if it exists. If it does not, then the default management scope is the peer domain scope if it exists. If it does not, then the management scope is the local scope. For more information, refer to the **rmaudrec** command man page and "How do I determine the target nodes for a command?" on page 75.

For detailed syntax information on the **Isaudrec** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Deleting entries from the audit log using the IBM.AuditLog resource's RetentionPeriod and MaxSize attributes:

In addition to being able to explicitly delete audit log entries using the **rmaudlog** command, you can also set certain attributes of the IBM.AuditLog resource that represents the audit log, so that RMC will automatically delete records from the audit log. These attributes are:

- the **RetentionPeriod** attribute which determines how many days RMC should keep records in the audit log. Records older than the number of days indicated are automatically deleted by RMC. If the **RetentionPeriod** attribute value is set to 0, this indicates that audit log records should not ever be automatically deleted based on their age.
- the **MaxSize** attribute which determines the maximum size (in Megabytes) of the audit log. If the size of the audit log exceeds the size indicated, RMC will automatically remove the oldest records until the size of the audit log is smaller than the indicated limit. The default size limit of the audit log is 1 Megabyte.

To list the current attribute settings, use the **lsrsrc** command (described in more detail in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*). To list the attribute settings for the IBM.AuditLog instance that represents the the ERRM audit log, use the selection string `-s 'Name == "ERRM"'`. For example:

```
lsrsrc -s 'Name == "ERRM"' IBM.AuditLog
```

This selection string is necessary since other subsystems may have their own audit logs. The preceding command will return output similar to the following.

```
Resource Persistent Attributes for: IBM.AuditLog
resource 1:
```

```
    Name           = "ERRM"
    MessageCatalog = "IBM.ERrm.cat"
    MessageSet     = 1
    DescriptionId  = 38
    DescriptionText = "This subsystem is defined by ERRM for recording significant event information."
    RetentionPeriod = 0
    MaxSize        = 1
    SubsystemId    = 1
    NodeNameList   = {"c175n06.ppd.pok.ibm.com"}
```

Included in this output are the attribute settings for the **RetentionPeriod** and **MaxSize** attributes. The **RetentionPeriod** attribute is set to 0; this indicates that RMC should not automatically delete records based on their age. The **MaxSize** attribute is set to 1; RMC will automatically delete the oldest records from the audit log when the audit log size exceeds 1 Megabyte.

To change these settings, use the **chrsrc** command. For example, to specify that RMC should automatically delete records that are over a day old, you would set the **RetentionPeriod** attribute as follows:

```
chrsrc -s 'Name == "ERRM"' IBM.AuditLog RetentionPeriod=3
```

To increase the maximum size of the audit log to 10 Megabytes, you would enter:

```
chrsrc -s 'Name == "ERRM"' IBM.AuditLog MaxSize=10
```

Note: The default size limit of the audit log is 1 Megabyte, which will be an insufficient size for a large cluster. In a large cluster you will likely want to increase the audit log size as shown in the preceding example. If you do set the **MaxSize** attribute to increase the maximum size limit of the audit log, be sure to verify that the size of the file system containing the log (by default, the **/var** file system) has enough room to hold it. Since RSCT subsystems

make extensive use of the `/var` file system, it is also a good idea to monitor its size. To monitor the `/var` file system, you can use the predefined condition `/var` space used provided by the File System Resource Manager. If you are a Cluster Systems Management (CSM) customer, you can also use the predefined condition `AnyNodeVarSpaceUsed` provided by the Domain Management Server Resource Manager. The Domain Management Server Resource Manager is only provided as part of CSM. The `AnyNodeVarSpaceUsed` condition monitors the `/var` file system on all nodes of the management domain.

For more information on the `lsrsrc` and `chrsrc` commands, refer their online man pages. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Advanced resource monitoring

As described in “Basic resource monitoring” on page 84, many predefined conditions and responses are provided by the various resource managers on your system. These predefined conditions and responses are provided as an administrative convenience. As described in “Creating a condition/response association” on page 88, you can use them to create condition/response associations for monitoring. However, the predefined conditions and responses may not always meet your needs. This section describes:

- how to create your own conditions that can then be linked with one or more responses and monitored by RMC. If the condition you wish to monitor is similar to one of the predefined conditions available on your system, this section shows you how you can copy the existing condition, and modify it as needed. If none of the existing conditions are similar to the condition you want to monitor, this section also shows how you can create a condition from scratch. This involves identifying the attribute you want to monitor for one or more resource of a particular resource class. Since persistent attributes are generally unchanging, you will usually monitor a dynamic attribute. If none of the dynamic attributes provided by the resource managers contains the value you want to monitor, this section also describes how you can create a *sensor* — a command to be run by RMC to retrieve the value you want to monitor. For more information, refer to “Creating, modifying and removing conditions.”
- how to create your own responses that can then be linked with conditions. This section describes the predefined response scripts that you can use in your responses. It also describes how you can create your own response scripts. For more information, refer to “Creating, modifying, and removing responses” on page 112.

While this section does discuss how to create conditions and responses, be aware that, to be effective, you need to link the conditions and responses together and start monitoring. These tasks are described in “Creating a condition/response association” on page 88 and “Starting condition monitoring” on page 90. For detailed syntax information on any the commands described in this section, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating, modifying and removing conditions

There are three commands you can use to manipulate conditions. You can:

- Create a new condition using the `mkcondition` command.

- Modify a condition using the **chcondition** command.
- Remove a condition using the **rmcondition** command.

Before we discuss these commands, it is important that you understand the basic attributes of a condition. In “Listing conditions” on page 85, we discuss the **lscondition** command that enables you to list conditions that are available. This command lists the predefined conditions we provide, as well as any you define. Specifying the name of a condition as a parameter to the **lscondition** command returns detailed information about the condition. For example, entering this command:

```
lscondition "/var space used"
```

Returns the following information about the predefined condition "/var space used".

Displaying condition information:

```
condition 1:
  Name           = "/var space used"
  Node           = "c175n06.ppd.pok.ibm.com"
  MonitorStatus  = "Not monitored"
  ResourceClass  = "IBM.FileSystem"
  EventExpression = "PercentTotUsed > 90"
  EventDescription = "An event will be generated when more than 90 percent
of the total space in the /var directory is in use."
  RearmExpression = "PercentTotUsed < 75"
  RearmDescription = "The event will be rearmed when the percent of the sp
ace used in the /var directory falls below 75 percent."
  SelectionString = "Name == \"/var\"
  Severity       = "i"
  NodeNames      = {}
  MgtScope       = "1"
```

It is important to understand the information contained in this output, because you can set many of these values using the various flags of the **mkcondition** and **chcondition** commands.

Table 24. Explanation of **lscondition** command output

This line of the lscondition command output:	Indicates:	Notes
Name = "/var space used"	The name of the condition. In this case "/var space used".	Specified as a parameter of the mkcondition and chcondition commands.
Node = "c175n06.ppd.pok.ibm.com"	The node on which the condition is defined. This is important, because, when you create a condition/response association, both the condition and the response must reside on the same node. In this case, the "/var space used" condition is defined on the node "c175n06.ppd.pok.ibm.com". This node information is provided only if the management scope is a peer domain scope or a management domain scope.	By default, will be the node where the mkcondition command runs. Can be explicitly specified using the mkcondition command's -p flag.
MonitorStatus = "Not monitored"	Whether or not the condition is being monitored. In this case, it is not.	See “Starting condition monitoring” on page 90 and “Stopping condition monitoring” on page 91.
ResourceClass = "IBM.FileSystem"	The resource class monitored by this condition. This will be the resource class that contains the attribute used in the event expression and, optionally, the rearm event expression. In this case, the resource class is the file system resource class (which contains the PercentTotUsed dynamic attribute used in the event expression and rearm event expressions).	Specified by the -r flag of both the mkcondition and chcondition commands.

Table 24. Explanation of **Iscondition** command output (continued)

This line of the Iscondition command output:	Indicates:	Notes
EventExpression = "PercentTotUsed > 90"	<p>The event expression used in monitoring the condition. Once you link the condition with one or more responses (as described in "Creating a condition/response association" on page 88), and start monitoring (as described in "Starting condition monitoring" on page 90), RMC will periodically poll the resource class to see if this expression (in this case "PercentTotUsed > 90") tests true. If it does test true, RMC will execute any response scripts associated with the condition's linked response(s).</p> <p>An event expression includes an attribute, a mathematical comparison symbol, and a constant.</p> <p>This particular expression uses the PercentTotUsed dynamic attribute which indicates the percentage of space used in a file system. When the file system is over 90 percent full, RMC generates an event, thus triggering any linked responses.</p>	Specified by the -e flag of both the mkcondition and chcondition commands.
EventDescription = "An event will be generated when more than 90 percent of the total space in the /var directory is in use."	A description of the event expression.	Specified by the -d flag of both the mkcondition and chcondition commands.
RearmExpression = "PercentTotUsed < 75"	<p>The rearm event expression. Once the event expression tests true, RMC will not test the event expression condition again until the rearm expression tests true. When this particular condition is monitored, for example, RMC will periodically poll the file system resource class to determine if the expression the test the event expression "PercentTotUsed > 90" is true. If it does, the linked responses are triggered, but, because there is a rearm event specified, RMC will then no longer test if "PercentTotUsed > 90" is true. If it did, the linked responses would be triggered every time RMC polled the file system resource class until the percentage of space used in the file system fell below 90 percent. If a linked response was to broadcast the information to all users who are logged in, the repeated broadcasts of the known problem would be unnecessary. Instead of this, the event expression testing true causes RMC to start testing the rearm event expression instead. Once it tests true, the condition is rearmed; in other words, the event expression is again tested. In this case, the condition is rearmed when the file system is less that 75 percent full.</p> <p>It is important to note that many conditions do not specify a rearm expression. When this is the case, the event expression will continue to be tested event after it tests true.</p>	Specified by the -E flag of both the mkcondition and chcondition commands.
RearmDescription = "The event will be rearmed when the percent of the space used in the /var directory falls below 75 percent."	A description of the rearm event expression.	Specified by the -D flag of both the mkcondition and chcondition commands.
SelectionString = "Name == \"/var\""	A selection string. This is an expression that determines which resources in the resource class are monitored. If a condition does not have selection string, then the condition would apply to all resources in the class. For example, if this condition did not have a selection string, the event expression would be tested against all file system resources in the file system resource class, and an event would occur if any of the file systems were over 90 percent full. However, since this selection string is defined, the condition applies only to the /var file system. The selection string can filter the resource class using any of its persistent attributes. In this case, that resource class is filtered using the Name attribute. Expressions in RMC are discussed in more detail in "Using expressions to specify condition events and command selection strings" on page 130.	Specified by the -s flag of both the mkcondition and chcondition commands.

Table 24. Explanation of **lscondition** command output (continued)

This line of the lscondition command output:	Indicates:	Notes
Severity = "i"	The severity of the condition. In this case, the condition is informational.	Specified by the -S flag of both the mkcondition and chcondition commands.
NodeNames = {}	The host names of the nodes where the condition is to be monitored. No hosts are named in this case. All nodes in the management scope will be monitored. For more information, refer to "What is a condition's monitoring scope?" on page 72.	Specified by the -n flag of both the mkcondition and chcondition commands.
MgtScope = "l"	The RMC scope in which the condition is monitored. In this case, the scope is the local node only. For more information, refer to "What is a condition's monitoring scope?" on page 72.	Specified by the -m flag of both the mkcondition and chcondition commands.

Creating a condition

To create a condition, you use the **mkcondition** command. Before creating a condition from scratch, you should make sure that it is truly necessary. In other words, first check to see if any of the predefined conditions is already set up to monitor the event you are interested in. For instructions on listing the conditions already available on your system, refer to "Listing conditions" on page 85. If you have additional resource managers provided by other products, such as the Cluster Systems Management (CSM) product which provides the Domain Management Server resource manager, refer to that product's documentation for information on any additional predefined conditions. If you are lucky, there is already a predefined condition that will monitor either the exact event you are interested in, or an event very similar.

If:	Then:
there is a predefined condition that exactly suits your needs	you do not need to perform this advanced task; instead, refer to "Creating a condition/response association" on page 88 and "Starting condition monitoring" on page 90.
there is a predefined condition very similar to the event you want to monitor	you can use the mkcondition command's -c flag to copy the existing condition, modifying only what you want to change to suit your needs. Refer to "Creating a condition by copying an existing one" on page 102 for more information.
there is no predefined condition that is similar to the event you want to monitor	you will need to define the condition completely from scratch. You will need to examine the available resource managers to see if any of them define an attribute containing the value you want to monitor. If none of them do, you can extend RMC by creating a <i>sensor</i> — a command to be run by RMC to retrieve the value you want to monitor. Refer to "Creating a condition from scratch" on page 104.

Targeting Node(s):

The **mkcondition** command is affected by the environment variables

CT_CONTACT and CT_MANAGEMENT_SCOPE. The CT_CONTACT environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The CT_MANAGEMENT_SCOPE indicates the management scope — either local scope, peer domain scope, or management domain scope. The **mkcondition** command's **-p** flag, if specified, indicates the name of a node where the condition is defined. This must be a node within the management scope for the local node (or the node indicated by the CT_CONTACT environment variable).

If the CT_MANAGEMENT_SCOPE environment variable is not set, and the **-p** flag is used, this command will attempt to set the CT_MANAGEMENT_SCOPE environment variable to the management scope that contains the node specified on the **-p** flag. In this case, the specified node should be in the management domain or peer domain of the local node (or the node indicated by the CT_CONTACT environment variable).

If using the **mkcondition** command on a CSM management server, do not specify the **-p** flag if you want the condition to be defined on the management server.

For more information, refer to the **mkcondition** command man page and “How do I determine the target nodes for a command?” on page 75.

Creating a condition by copying an existing one: If there is an existing condition very similar to the event you want to monitor, you can use the **mkcondition** command's **-c** flag to copy the existing condition, modifying only what you want to change to suit your needs. For example, say you want to monitor the **/var** file system, and generate an event when the file system is 85 percent full. Using the **lscondition** command, as described in “Listing conditions” on page 85, shows that there is a predefined condition named “/var space used”. To get detailed information about this predefined condition, you enter the following command:

```
lscondition "/var space used"
```

Which causes the following information to be output.

Displaying condition information:

```
condition 1:
  Name           = "/var space used"
  Node           = "c175n06.ppd.pok.ibm.com"
  MonitorStatus  = "Not monitored"
  ResourceClass  = "IBM.FileSystem"
  EventExpression = "PercentTotUsed > 90"
  EventDescription = "An event will be generated when more than 90 percent
of the total space in the /var directory is in use."
  RearmExpression = "PercentTotUsed < 75"
  RearmDescription = "The event will be rearmed when the percent of the sp
ace used in the /var directory falls below 75 percent."
  SelectionString = "Name == \"/var\"
  Severity       = "i"
  NodeNames      = {}
  MgtScope       = "1"
```

This **lscondition** output (described in detail in Table 24 on page 99) shows that the predefined condition “/var space used” is very similar to what you are looking for; the only difference is that it triggers an event when the **/var** file system is 90 percent full instead of 85 percent full. While you could just modify the “/var space used” condition itself (as described in “Modifying a condition” on page 111), you

think it's best to leave this predefined condition as it is, and instead copy it to a new condition. The following **mkcondition** command creates a condition named `"/var space 85% used"` that copies the `"/var space used"` condition, modifying its event expression.

```
mkcondition -c "/var space used" -e "PercentTotUsed > 85" -d "An event
will be generated when more than 85 percent" "/var space 85% used"
```

In the preceding command:

- `-c "/var space used"` indicates that you want to use the `"/var space used"` condition as a template for the new condition.
- `-e "PercentTotUsed > 85"` modifies the condition's event expression.
- `-d "An event will be generated when more than 85 percent"` modifies the condition's event description to reflect the new event expression.
- `"/var space 85% used"` is the name for the new condition.

After running the above command, the `"/var space 85% used"` condition will be included in the list generated by the **lscondition** command, showing that the condition is available for use in a condition/response associated. To see the new condition's detailed information, enter:

```
lscondition "/var space 85% used"
```

Which will display the following output:

Displaying condition information:

```
condition 1:
  Name           = "/var space 85% used"
  Node           = "c175n06.ppd.pok.ibm.com"
  MonitorStatus  = "Not monitored"
  ResourceClass  = "IBM.FileSystem"
  EventExpression = "PercentTotUsed > 85"
  EventDescription = "An event will be generated when more than 85 percent"
  RearmExpression = "PercentTotUsed < 75"
  RearmDescription = "The event will be rearmed when the percent of the spa
ce used in the /var directory falls below 75 percent."
  SelectionString = "Name == \"\/var\""
  Severity       = "i"
  NodeNames     = {}
  MgtScope      = "1"
```

Notice that the new condition is an exact copy of the `"/var space used"` condition except for the modifications specified on the **mkcondition** command.

If you want to prevent user modification or removal of this condition, you could lock it. For more information, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

If the condition you want to copy is defined on another node of a peer domain or management domain, you can specify the node name along with the condition. For example:

```
mkcondition -c "/var space used":nodeA -e "PercentTotUsed > 85" -d "An event
will be generated when more than 85 percent" "/var space 85% used"
```

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the `CT_MANAGEMENT_SCOPE` environment variable) for the local node or the node specified by the `CT_CONTACT` environment variable (if it is

set). For more information, refer to the **mkcondition** command man page and “How do I determine the target nodes for a command?” on page 75.

This next example illustrates two other flags of the **mkcondition** command. The **-E** flag specifies a rearm expression, and the **-D** flag modifies the rearm expression description.

```
mkcondition -c "/var space used" -E "PercentTotUsed < 70" -D "The event will be rearmed when the percent of the space used in the /var directory falls below 70 percent." "modified /var space used"
```

This next example illustrates the flags of the **mkcondition** command that you can use to set the condition's monitoring scope. The condition's monitoring scope refers to the node or set of nodes where the condition is monitored. Although a condition resource is defined on a single node, its monitoring scope could be the local node only, all the nodes of a peer domain, select nodes of a peer domain, all the nodes of a management domain, or select nodes of a management domain. If the monitoring scope indicates nodes of a peer domain or management domain, the node on which the condition resource is defined must be part of the domain. The monitoring scope is, by default, the local node on which the condition resource resides. To specify a peer domain or management domain, you use the **-m** option. The setting **-m p** indicates a peer domain monitoring scope, and **-m m** indicates a management domain monitoring scope. (The **-m m** option is allowed only if you are defining the condition on the management server of the management domain.) To further refine this monitoring scope, you can use the **-n** option to specify select nodes in the domain. In this next example, we copy the `"/var space used"` condition, but modify its monitoring scope to certain nodes in a peer domain.

```
mkcondition -c "/var space used" -m p -n nodeA,nodeB "/var space used nodeA,nodeB"
```

Finally, let's say you want a condition that generates an event when the `/usr` file system is 90 percent full. You could again copy the `"var space used"` condition, this time using the **mkcondition** command's **-s** option to specify a different selection string expression. (Since the rearm expression description mentions the `/var` file system, we will modify that as well.)

```
mkcondition -c "/var space used" -s "Name == \"/usr\"" -D "The event will be rearmed when the percent of the space used in the /usr directory falls below 75 percent." "/usr space used"
```

In the above example, modifying the event expression was fairly straightforward. Expressions in RMC are discussed in more detail in “Using expressions to specify condition events and command selection strings” on page 130. Here it suffices to say that the syntax of the selection string is similar to an expression in the C programming language or the *where* clause in SQL. In this case, the condition uses the expression `"Name == \"/usr\""`, so that the condition applies only to resources in the class whose Name persistent attribute value is `/usr`.

Creating a condition from scratch: Usually, the predefined conditions we provide will meet your monitoring needs with, at most, minor modifications. However, if no existing condition is similar to the only you want to create, you need to define the condition completely. To do this, you will need to understand the basic attributes of a condition. Refer to table Table 24 on page 99 which describes the attributes of a condition using the predefined condition `/var space used` as an example.

Once you understand the information contained in Table 24 on page 99, you can use the following steps to create a condition. There is a significant amount of information you'll need to provide to the **mkcondition** command when defining a condition from scratch. The steps that follow are ordered so that you can carefully

consider the purpose and implications of each piece of information you need to supply. The steps culminate in actually issuing the **mkcondition** command:

1. **Identify the attribute you want to monitor.** While resource classes define both persistent and dynamic attributes, it is usually dynamic attributes that are monitored. This is because a persistent attribute is less likely to change (and then only by someone explicitly resetting it). An instance of the Processor resource class, for example, has a persistent attribute **ProcessorType** that identifies the type of processor. It would be pointless to monitor this attribute; it's not going to change. Dynamic attributes, however, track changing states. An instance of the Processor resource class, for example, has a dynamic attribute **OpState** that indicates whether the operational state of the processor is online or offline.

For monitoring data, the key resource managers are the Host resource manager and the File System resource manager. These two resource managers contain the resource classes whose dynamic attributes reflect variables to monitor.

- The Host resource manager enables you monitor system resources for individual machines. In particular, it enables you to monitor operating system load and status.
- The File System resource manager enables you to monitor file systems. In particular, it enables you to monitor the percentage of disk space and the percentage of i-nodes used by individual file systems.

For more information on the resource managers and the resource classes they define, refer to “What resource managers are provided with RSCT?” on page 65.

If you have additional resource managers provided by other products, such as the Cluster Systems Management (CSM) product which provides the Domain Management Server resource manager, refer to that product's documentation for information on additional resource classes and what attributes they enable you to monitor. You can also examine the available resource classes and attributes using RMC commands (such as the **lsrsrc** command). Refer to “How does RMC and the resource managers enable you to manage resources?” on page 70 for more information on RMC commands. For complete syntax information on the commands, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Note: If, after examining the dynamic attributes provided by the available resource managers, you determine that there are none that contain the value you want to monitor, you can extend RMC by creating a *sensor*. A sensor is a command to be run by RMC (at specified intervals and/or when you explicitly request for it to be run) to retrieve the value you want to monitor. Refer to “Creating event sensor commands for monitoring” on page 109 for more information.

For example, let's say you are interested in monitoring the operational state of processors, and would like the system to notify you if a processor goes offline. (There is, in fact, a predefined condition designed to monitor this, but for the sake of this discussion, we'll assume it was accidentally removed.) To see if there are any resource classes that represent processors, you can issue the **lsrsrc** or **lsrsrcdef** command without any parameters or flags.

```
lsrsrc
```

This displays output similar to the following:

```

class_name
"IBM.Association"
"IBM.ATMDevice"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EthernetDevice"
"IBM.EventResponse"
"IBM.FDDIDevice"
"IBM.Host"
"IBM.FileSystem"
"IBM.PagingDevice"
"IBM.PhysicalVolume"
"IBM.Processor"
"IBM.Program"
"IBM.TokenRingDevice"
...

```

The IBM.Processor resource class sounds promising. For details on the resources in this class, enter the following **lsrsrc** command. The **-A d** instructs the command to list only dynamic attributes.

```
lsrsrc -A d IBM.Processor
```

This displays output similar to the following:

```

Resource Dynamic Attributes for: IBM.Processor
resource 1:
    PctTimeUser   = 0.0972310851777207
    PctTimeKernel = 0.446023453293117
    PctTimeWait   = 0.295212932824663
    PctTimeIdle   = 99.1615325287045
    OpState       = 1
resource 2:
    PctTimeUser   = 0.0961145070660594
    PctTimeKernel = 0.456290452125732
    PctTimeWait   = 0.30135492264433
    PctTimeIdle   = 99.1462401181639
    OpState       = 1
resource 3:
    PctTimeUser   = 0.102295524109806
    PctTimeKernel = 0.475051721639257
    PctTimeWait   = 0.316998288621668
    PctTimeIdle   = 99.1056544656293
    OpState       = 1
resource 4:
    PctTimeUser   = 0.0958503317766613
    PctTimeKernel = 0.452945804277402
    PctTimeWait   = 0.30571948042647
    PctTimeIdle   = 99.1454843835195
    OpState       = 1

```

The preceding output shows us that there are five dynamic attributes. While you can get detailed information about these attributes using the **lsrsrcdef** command (as described in “Displaying attribute definition information for a resource or a resource class” on page 83), the names are fairly self-explanatory. The **OpState** attribute monitors whether the processor is online or offline, while the others represent the percentage of time the processor spends in various states. (Of course, the Host resource manager provides predefined conditions for all of these dynamic attributes, so you would not have to create a condition from scratch and could instead either use the predefined conditions as is, or follow the instructions in “Creating a condition by copying an existing one” on page 102. For the sake of this discussion, we’ll assume no predefined conditions are available.)

Now that we've found a dynamic attribute (OpState) that contains the information we want to monitor, we can move on to the next step.

2. **Design an event expression that will test the attribute for the condition of interest.** Once you have identified the attribute that contains the information you want to monitor, you need to design the event expression you will supply to the **mkcondition** command. An event expression includes the attribute, a mathematical comparison symbol, and a constant. RMC will periodically poll the resource class to determine if this expression is true. If the expression does test true, RMC will execute any response scripts associated with the condition's linked responses.

RMC keeps track of the previously observed value of an attribute. If an event expression appends an attribute name with "@P", this refers to the previously observed value of the attribute. An event expression might use this capability to compare the currently observed value of the attribute with its previously-observed value. For example, the following event expression, if specified on a condition, would trigger an event if the average number of processes on the run queue has increased by 50% or more between observations:

```
(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)
```

Expressions in RMC are described in more detail in "Using expressions to specify condition events and command selection strings" on page 130.

In our example, we want to create a condition that creates an event when a processor goes offline. We've found that the OpState dynamic attribute of the Processor resource class contains this information. If the value of OpState is 1, the processor is online. The expression "OpState != 1" will therefore test true if the processor is offline.

3. **Design a rearm event expression if you determine that one is necessary.** To determine whether a rearm event expression is needed in this condition, consider how the condition will behave later when you have started monitoring it. In our example, RMC will periodically poll the Processor resource class to determine if the expression "OpState != 1" tests true. If it does, the event occurs, triggering the condition's linked responses. If there is a rearm expression defined, RMC will, the next time it polls the Processor resource class, test the rearm expression. It will continue to test the rearm expression, until it tests true; only then will RMC resume testing the event expression. If the condition has no rearm expression, then RMC will continue to test the event expression each time it polls the Processor resource class. The linked responses will be triggered each time the event expression is evaluated until the processor is brought back online. Since the linked response might be send e-mail to root or notify everyone on the system, you probably only want this happening once when the processor is first detected offline. We will use "OpState == 1" as our rearm expression; the condition will be rearmed only after the processor is detected to be back online.
4. **Determine the condition's monitoring scope.** If you are on a cluster of nodes configured into management and/or peer domains, the condition's monitoring scope refers to the node or set of nodes where the condition is monitored. Although a condition resource is defined on a single node, its monitoring scope could be the local node only, all the nodes of a peer domain, select nodes of a peer domain, all the nodes of a management domain, or select nodes of a management domain. The monitoring scope is, by default, the local node on which the condition resource resides. To specify a peer domain or management domain, you use the **-m** option. The setting **-m p** indicates a peer domain monitoring scope, and **-m m** indicates a management domain monitoring scope. (The **-m m** option is allowed only if you are defining the

condition on the management server of the management domain.) To further refine this monitoring scope, you can use the **-n** option to specify select nodes in the domain.

In our example, we'll just monitor the local node on which the condition is defined. Since this is the default behavior, we will not need to use the **-m** flag.

For more information on domains in a cluster, refer to "What are management domains and peer domains?" on page 1. For more information on the **-m** flag, refer to the **mkcondition** command's online man page. Detailed syntax information is also available in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

5. **Design a selection string if you determine that one is necessary.** By default, the condition will apply to all resources in the class. However, a selection string expression, if provided, will filter the resource class so that the condition will apply only to resources that match the expression. The event expression can filter the resource class using any of its persistent attributes. To understand how this works, let's look at the resources in the Processor resource class. The following **lsrsrc** command lists each resource in the Processor resource class. The **-A p** instructs the command to list only the persistent resource attributes of the resources.

```
lsrsrc -A p IBM.Processor
```

The following output is returned.

```
Resource Persistent Attributes for: IBM.Processor
resource 1:
    Name           = "proc3"
    NodeNameList   = {"c175n06.ppd.pok.ibm.com"}
    ProcessorType  = "PowerPC_604"
resource 2:
    Name           = "proc2"
    NodeNameList   = {"c175n06.ppd.pok.ibm.com"}
    ProcessorType  = "PowerPC_604"
resource 3:
    Name           = "proc1"
    NodeNameList   = {"c175n06.ppd.pok.ibm.com"}
    ProcessorType  = "PowerPC_604"
resource 4:
    Name           = "proc0"
    NodeNameList   = {"c175n06.ppd.pok.ibm.com"}
    ProcessorType  = "PowerPC_604"
```

Here we can see that there are four processors that, by default, will all be monitored by the condition. For our example condition, this is the behavior we are looking for. If for some reason we wanted to monitor only the processor named "proc3", we would use the selection string "Name = "proc3"".

6. **Determine the severity of the event.** Should the event be considered a critical error, a warning, or merely informational. We'll consider our example condition informational.
7. **Create the condition using the mkcondition command.** Now it's time to put it all together. The following **mkcondition** command defines our condition.

```
mkcondition -r IBM.Processor -e "OpState != 1" -d "processor down"
-E "OpState == 1" -D "processor online" -S i "new condition"
```

In the preceding command:

- the **-r** flag specifies the resource class containing the attribute to be monitored.
- the **-e** flag specifies the event expression.

- the **-d** flag specifies a short description of the event expression.
- the **-E** flag specifies the rearm expression.
- the **-D** flag specifies a short description of the event expression.
- the **-S** flag specifies the severity of the condition.

If you wanted to prevent user modification or removal of this condition, you could lock it. For more information, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

For detailed syntax information on the **mkcondition** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating event sensor commands for monitoring: When none of the attributes of the available resource classes contain the value you are interested in monitoring, you can extend the RMC system by creating a *sensor*. A *sensor* is merely a command that the RMC subsystem runs to retrieve one or more user-defined values. You can define a sensor to be run at set intervals and/or you can run it explicitly. The sensor is essentially a resource that you add to the Sensor resource class of the Sensor resource manager. The values returned by the script are dynamic attributes of that resource. You can then create a condition to monitor these dynamic attributes that you have defined.

To create a sensor and condition to monitor a dynamic attribute it defines:

1. **Identify a variable value that none of the existing resource managers currently return.** For example, say you want to monitor the number of users logged on to the system. This is a variable that none of the existing resource managers define. Since there is no existing attribute that contains the value, you’ll need to create a sensor if you want to monitor this value.
2. **Create the sensor command script that RMC will run to retrieve the system value(s) of interest.** In our example, we said we wanted to monitor the number of users currently logged on to the system. This following script will retrieve this information:

```
#!/usr/bin/perl
my @output='who';
print 'Int32=scalar(@output), "\n";
exit;
```

When creating sensor command scripts, be aware of the following:

- The command should return the value it retrieves from the system by sending it to standard output in the form *attribute=value*. The *attribute* name used depends on the type of the value and is one of these: **String**, **Int32**, **Uint32**, **Int64**, **Uint64**, **Float32**, **Float64**, or **Quantum**. (If only the value is sent to standard output, the attribute name is assumed to be **String**.)
 - If the command returns more than one type of data, it should send a series of *attribute=value* pairs to standard output, separated by blanks (for example: `Int32=10 String="abcdefg"`).
3. **Add your sensor command to the RMC subsystem.** One you have created the sensor command script, you need to add it to the RMC subsystem so that RMC will execute the command at intervals to retrieve the value of interest. To do this, you create a sensor object using the **mksensor** command. When entering this command, you need to name the sensor you are creating and

provide the full path name of the sensor command script. For example, if our sensor command script is `/usr/local/bin/numlogins`, then we could create the sensor named **NumLogins** by entering:

```
mk sensor NumLogins /usr/local/bin/numlogins
```

As soon you create the sensor, RMC will periodically execute its associated script to retrieve the value. The value will be stored as a dynamic attribute of the Sensor resource. In our example, the number of users currently logged onto the system will be the value of the **NumLogins** resource's **Int32** dynamic attribute.

By default, RMC will execute the sensor command script at 60-second intervals. To specify a different interval, use the `-i` flag of the **mk sensor** command. For example, to specify that RMC should execute our **numlogins** script at five-minute (300-second) intervals, you would enter:

```
mk sensor -i 300 NumLogins /usr/local/bin/numlogins
```

In addition to any interval you set, you can also explicitly execute the sensor command using the **ref sensor** command. For example:

```
ref sensor NumLogins
```

The **ref sensor** command refreshes a sensor and is independent of, and in addition to, the refresh interval you set. If you prefer to only manually run the sensor using the **ref sensor** command, you can set the interval to 0. For example:

```
mk sensor -i 0 NumLogins /usr/local/bin/numlogins
```

When creating a sensor, be aware of the following:

- Since the sensor resource identifies the sensor command script using a full path name. Therefore, the sensor must be defined on the same node as the command script, or otherwise accessible to it (for example, in a shared file system).
- RMC will execute the sensor command script in the process environment of the user who invokes the **mk sensor** command. This user should therefore have the permissions necessary to run the command script. If the command script can only be run by the root user, then the root user must issue the **mk sensor** command.

4. **Create a condition to monitor a dynamic attribute of the sensor.** The **mk sensor** command creates a sensor resource of the Sensor resource class. The sensor command script associated with this resource is executed at set intervals and/or when you issue the **ref sensor** command. Any value returned by the script is stored as a dynamic attribute of the sensor resource. In our example, the sensor resource is named **NumLogins**, and (since its associated command script contains the statement `print 'Int32='scalar(@output), "\n";`) the value we're interested will be available in the **Int32** dynamic attribute. So the following condition will trigger an event if any users are logged into the system.

```
mk condition -r IBM.Sensor -e "Int32 != 0" -d "users logged in" "users online"
```

In addition to being able to create conditions based on the output of the sensor command script, be aware that the exit value of the script is stored in the Sensor resource's **ExitValue** attribute, and so you can also create a condition based on this.

For detailed syntax information on the **mk sensor** and **ref sensor** commands, refer to their online man pages. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*. This reference

also has information on the related sensor commands **lssensor** (list sensors), **chsensor** (modify a sensor), and **rmsensor** (remove sensor).

Modifying a condition

To modify a condition, you use the **chcondition** command. The **chcondition** command uses the same flags as the **mkcondition** command, so it is simply a matter of supplying the **chcondition** command with the name of the condition to change and any changes you want to make. For example, to modify the event expression and event description of the `"/var space used"` condition, you would use the **-e** and **-d** flags.

```
chcondition -e "PercentTotUsed > 85" -d "An event
will be generated when more than 85 percent" "/var space used"
```

To modify the rearm event expression and rearm description, you would use the **-E** and **-D** flags.

```
chcondition -E "PercentTotUsed < 70" -D "The event will be
rearmed when the percent of the space used in the /var directory falls below 70
percent." "/var space used"
```

To modify the condition's selection string expression, you would use the **-s** flag.

```
chcondition -s "Name == \"/usr\"/" "/var space used"
```

To rename a condition, you would use the **-c** flag. For example, the condition in the preceding example should probably not be called `"/var space used"` anymore, since the selection string has been modified so that the condition applies to the **/usr** file system. To change the name of this condition from **"/var space used"** to **"/usr space used"**, you would enter:

```
chcondition -c "/usr space used" "/var space used"
```

You will not be able to modify a condition that is locked. Instead, the **chcondition** command will generate an error informing you that the condition is locked. For more information on unlocking a condition so it can be modified, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

For detailed syntax information on the **chcondition** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing a condition

The **rmcondition** command enables you to remove a condition. For example:

```
rmcondition "/usr space used"
```

If the condition you have specified has linked responses, an error message will display and the condition will not be removed. To remove a condition even if it has linked responses, use the **-f** (force) flag. For example:

```
rmcondition -f "/usr space used"
```

If the condition you want to remove is defined on another node of a peer domain or management domain, you can specify the node name along with the condition. For example:

```
rmcondition "/usr space used":nodeA
```

You will not be able to remove a condition that is locked. Instead, the **rmcondition** command will generate an error informing you that the condition is locked. For more

information on unlocking a condition so it can be removed, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the CT_MANAGEMENT_SCOPE environment variable) for the local node or the node specified by the CT_CONTACT environment variable (if it is set). For more information, refer to the **rmcondition** command man page and “How do I determine the target nodes for a command?” on page 75.

For detailed syntax information on the **rmcondition** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating, modifying, and removing responses

There are three commands you can use to manipulate responses. You can:

- Create a new response using the **mkresponse** command.
- Modify a response using the **chresponse** command.
- Remove a response using the **rmresponse** command.

Before we discuss these commands, it is important that you understand the basic attributes of a response. In “Listing responses” on page 86, we discuss the **lsresponse** command that enables you to list responses that are available. This command lists the predefined responses we provide, as well as any you define. Specifying the name of a response as a parameter to the **lsresponse** command returns detailed information about the response. For example, entering this command:

```
# lsresponse "Informational notifications"
```

Returns the following information about the predefined response "Informational notifications".

Displaying response information:

```
ResponseName = "Informational notifications"
Node         = "c175n06.ppd.pok.ibm.com"
Action      = "Log info event"
DaysOfWeek  = 1-7
TimeOfDay   = 0000-2400
ActionScript = "/usr/sbin/rsct/bin/logevent /tmp/infoEvents"
ReturnCode  = -1
CheckReturnCode = "n"
EventType   = "b"
StandardOut = "n"
EnvironmentVars = ""
UndefRes    = "n"

ResponseName = "Informational notifications"
Node         = "c175n06.ppd.pok.ibm.com"
Action      = "E-mail root"
DaysOfWeek  = 2-6
TimeOfDay   = 0800-1700
ActionScript = "/usr/sbin/rsct/bin/notifyscript root"
ReturnCode  = -1
CheckReturnCode = "n"
```

```

EventType      = "b"
StandardOut    = "n"
EnvironmentVars = ""
UndefRes       = "n"

```

Each block of information in the preceding output represents a different action associated with the response. You can think of a response as a wrapper around the actions that can be performed when any condition linked with the response tests true. When such a condition event occurs, the response is triggered, and any number of its actions may then be executed. When adding an action to a response, you specify the day(s) of the week and hour(s) of the day when the action can execute. If the linked condition event occurs during a time when the action is defined to run, it will execute. Otherwise, the action will not execute. This enables the system to respond one way to an event during work hours, and another way outside work hours. The preceding command output, for example, shows that during work hours, the response action will be to e-mail root. Outside work hours, however, the response action is to merely log the information.

It is important to understand the information contained in the preceding output, because you can set many of these values using the various flags of the **mkresponse** and **chresponse** commands. Let's look at the information for one of the associated actions in more detail.

Table 25. Explanation of **lsresponse** command output

This line of the lsresponse command output:	Indicates:	Notes
ResponseName = "Informational notifications"	The name of the response. In this case "Informational notifications".	Specified as a parameter of the mkresponse and chresponse commands.
Node = "c175n06.ppd.pok.ibm.com"	The node on which the response is defined. This is important, because, when you create a condition/response association, both the condition and the response must reside on the same node. In this case, the "E-mail root off-shift" response is defined on the node "c175n06.ppd.pok.ibm.com". This node information is provided only if the management scope is a peer domain scope or a management domain scope.	By default, will be the node where the mkresponse command runs. Can be explicitly specified using the mkresponse command's -p flag.
Action = "E-mail root"	The name of this response action. This name describes what the action script does.	Specified by the -n flag of both the mkresponse and chresponse commands.
DaysOfWeek = 2-6	The days of the week that this action can execute. The days of the week are numbered from 1 (Sunday) to 7 (Saturday). This particular action will not execute on weekends. If the response is triggered on Saturday or Sunday, this response action will not run.	Specified by the -d flag of both the mkresponse and chresponse commands.
TimeOfDay = 0800-1700	The range of time during which the action can execute. This particular action will execute only during work hours (between 8 am and 5 pm). If the response is triggered outside of these hours, this response action will not run.	Specified by the -t flag of both the mkresponse and chresponse commands.
ActionScript = "/usr/sbin/rsct/bin/notifyevent root"	The full path to the script or command to run for this action. This particular script will e-mail the event information to root.	Specified by the -s flag of both the mkresponse and chresponse commands.

Table 25. Explanation of **lsresponse** command output (continued)

This line of the lsresponse command output:	Indicates:	Notes
ReturnCode = -1	The expected return code of the action script.	Specified by the -r flag of both the mkresponse and chresponse commands.
CheckReturnCode = "n"	Whether or not RMC compares the action script's actual return code to its expected return code. If RMC does make this comparison, it will write a message to the audit log indicating whether they match. If RMC does not make this comparison, it will merely write the actual return code to the audit log. For more information on the the audit log, refer to "Using the audit log to track monitoring activity" on page 93.	Implied by specifying an expected return code using the -r flag of both the mkresponse and chresponse commands.
EventType = "b"	Whether this action should be triggered for the condition's event, rearm event, or both the event and rearm event. This action applies to both the event and rearm event. If either the event expression or the rearm expression of a condition linked to this response tests true, this action can be triggered.	Specified by the -e flag of both the mkresponse and chresponse commands.
StandardOut = "n"	Whether standard output should be directed to the audit log. For more information on the audit log, refer to "Using the audit log to track monitoring activity" on page 93.	Specified by the -o flag of both the mkresponse and chresponse commands.
EnvironmentVars = ""	Environment variables that RMC should set prior to executing the action script. This enables you to create general-purpose action scripts that respond differently, or provide different information, depending on the environment variable settings. (In addition to any environment variables you define this way, also be aware that RMC sets many variables that the action script can use. For more information, refer to Table 27 on page 118.)	Specified by the -E flag of both the mkresponse and chresponse commands.
UndefRes = "n"	Indicates whether or not RMC should still execute the action script if the resource monitored by the condition becomes undefined.	Specified by the -u flag of both the mkresponse and chresponse commands.

The rest of this section describes how to create responses using the **mkresponse** and **chresponse** commands. The **mkresponse** command creates the response with, optionally, one action specification. To add additional actions to the response, you can then use the **chresponse** command. The **chresponse** command also enables you to remove an action from the response, or rename the response. This section also describes how to remove a response when it is no longer needed. To do this, you use the **rmresponse** command.

In addition to any responses you create, be aware that we provide predefined responses. These are described in Table 16 on page 73.

Creating a response

To create a response, you use the **mkresponse** command. Before creating one, however, you should first check to see if any of our predefined responses are suitable for your purposes. Refer to Table 16 on page 73. For instructions on listing the predefined responses available on your system, refer to "Listing responses" on page 86. If you are lucky, there is already a predefined response that does what you need. In that case, you do not need to preform this advanced task and can instead refer to "Creating a condition/response association" on page 88 and "Starting condition monitoring" on page 90.

Once you understand the information contained in Table 25 on page 113, you can use the following steps to create a response. Keep in mind that the **mkresponse** command enables you to define one action only. In fact, with the exception of the response name, the information you supply to this command describes the action. Once you have defined the response using the **mkresponse** command, you can add more actions to it using the **chresponse** command.

1. **Decide which action script, if any, should be triggered by the response.**

There are a number of predefined action scripts that you can associate with the action. You can also create your own action script and associate it with the action. In addition, information about the response occurring will be entered into the audit log. You do not need to associate an action script with the action; if you do not, the response information will still be entered into the audit log.

The predefined action scripts are located in the directory **/usr/sbin/rsct/bin/** and are described in the following table.

Table 26. Predefined Response Scripts

Script	Description
displayevent	Available on Linux nodes only. Sends a message about the event to a specified X-window display.
logevent	Logs information about the event to a specified log file. The name of the log file is passed as a parameter to the script. This log file is not the audit log; it is a file you specify.
msgevent	Available on Linux nodes only. Sends information about the event to a specified user's console.
notifyevent	E-mails information about the event to a specified user ID. This user ID can be passed as a parameter to the script, or else is the user who ran the command.
snmpevent	Sends a Simple Network Management Protocol (SNMP) trap to a host running an SNMP event.
wallevent	Broadcasts the event information to all users who are logged in.

Note: The **/usr/sbin/rsct/bin/** directory also contains variations of three of these scripts called **elogevent**, **enotifyevent**, and **ewallevent**. These have the same functionality as the scripts outlined in the preceding table; the only difference is that they always return messages in English, while the scripts outlined in the table return messages based on the local language setting.

In addition to our predefined scripts which, as you can see from the preceding table, perform general-purpose actions, you can also create your own action scripts. One reason you might do this is to create a more targeted response to an event. For example, you might want to write a script that would automatically delete the oldest unnecessary files when the **/tmp** file system is 90 percent full. For more information, refer to "Creating new response scripts" on page 117.

If you decide to use one of our predefined action scripts, be sure you understand exactly what the script will do. For more information on a script, refer to the script's online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Whether you choose one of our predefined scripts or one you create, you will specify it to using the **mkresponse** command's **-s** flag. You'll need to provide the full path name of the script and any parameters you need or want to pass it.

For example, let's say you want to use the log event script to log the event information to the file **/tmp/EventLog**. The specification would be:

```
-s "/usr/sbin/rsct/bin/logevent /tmp/EventLog"
```

2. **Decide on the days/hours during which this action can be run.** Some actions may only be appropriate or desired during work hours, some may only be desired outside work hours. Often a response will have multiple actions, each designed for different days or times. For example, one action might be defined to run only during work hours and would notify you by e-mail about an error. Another action on the same response might run only outside work hours and would merely log the error to a file.

The **mkresponse** command's **-d** option specifies the days of the week that the command can execute. The days are numbered from 1 (Sunday) to 7 (Saturday). You can specify either a single day (7), multiple days separated by a plus sign (1+7), or a range of days separated by a hyphen (2-6).

Using the **mkresponse** command's **-t** flag, you can specify the range of time during which the command can run. The time is specified in a 24-hour format, where the first two digits represent the hour and the second two digits are the minutes. The start time and end time are separated by a hyphen. So, for example, if we wanted the action to run only during work hours (Monday through Friday, 8 am to 5 pm), the specification would be:

```
-d 2-6 -t 0800-1700
```

You can also specify different times for different days by making multiple specifications with the **-d** and **-t** flags. The number of day parameters must match the number of time parameters. For example, if you wanted the action to be used anytime Saturday and Sunday, but only between 8 am and 5 pm on the weekdays, you would use the following specification.

```
-d 1+7,2-6 -t 0000-2400,0800-1700
```

3. **Decide if this action should apply to the condition event, condition rearm event, or both.** You specify this using the **-e** flag with the setting **a** (event only), **r** (rearm event only), or **b** (both event and rearm event). For example, if you want the action to be executed in response the condition event only, the specification would be:

```
-e a
```

4. **Create the response using the mkresponse command.** Once you understand the action you want to define, you can enter the **mkresponse** command with all the appropriate option settings. Use the **-n** flag to specify the action name, and pass the response name as a parameter to the command. For example:

```
mkresponse -n LogAction -s /usr/sbin/rsct/bin/logevent /tmp/EventLog  
-d 1+7,2-6 -t 0000-2400,0800-1700 -e a "log info to /tmp/EventLog"
```

The preceding command creates a response named "log info to /tmp/EventLog". If you wanted to prevent user modification or removal of this response, you could lock it. For more information, refer to "Locking and unlocking conditions, responses, and condition/response links" on page 128.

To add additional actions to a response, use the **chresponse** command, as described in "Modifying a response" on page 120.

Targeting Node(s):

The **mkresponse** command is affected by the environment variables **CT_CONTACT** and **CT_MANAGEMENT_SCOPE**. The **CT_CONTACT** environment variable indicates a node whose RMC daemon will carry out the command request (by default, the local node on which the command is issued). The **CT_MANAGEMENT_SCOPE** indicates the management scope — either local scope, peer domain scope, or management domain scope.

The **mkresponse** command's **-p** flag, if specified, indicates the name of a node where the response is defined. This must be a node within the management scope for the local node (or the node indicated by the `CT_CONTACT` environment variable).

If the `CT_MANAGEMENT_SCOPE` environment variable is not set, and the **-p** flag is used, this command will attempt to set the `CT_MANAGEMENT_SCOPE` environment variable to the management scope that contains the node specified on the **-p** flag. In this case, the specified node should be in the management domain or peer domain of the local node (or the node indicated by the `CT_CONTACT` environment variable).

If using the **mkresponse** command on a CSM management server, do not specify the **-p** flag if you want the condition to be defined on the management server.

For more information, refer to the **mkresponse** command man page and "How do I determine the target nodes for a command?" on page 75.

For detailed syntax information on the **mkresponse** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Creating new response scripts: The predefined response scripts we provide are general purpose ways of notifying users about an event, or else logging the event information to a file. In addition to these general-purpose scripts, you might want to write your own scripts that provide more specific responses to events. You might want to do this to create an automatic recovery script that would enable RMC to solve a simple problem automatically. For example when the `/tmp` directory is over 90 percent full, you could have RMC run a script to automatically delete the oldest unnecessary files in the `/tmp` directory. Another reason you might want to create your own scripts is to tailor system responses to better suit your particular organization. For example, you might want to create a script that calls your pager when a particular event occurs.

If you want to create your own response scripts, it pays to examine the existing scripts we provide (as described in Table 26 on page 115). These scripts are located in the directory `/usr/bin/rsct/bin`, and can be useful as templates in creating your new scripts, and also illustrate how the script can use ERRM environment variables to obtain information about the event that triggered its execution. For example, say you wanted to create a script that called your pager when particular events occur. You might want to use our predefined script **wallevent** as a template in creating your new script. This predefined script uses the **wall** command to write a message to all users who are logged in. You could make a copy of this program, and replace the **wall** command with a program to contact your pager.

Note: Because our predefined responses use the predefined response scripts, do not modify the original scripts in `/usr/bin/rsct/bin`. In you want to use an existing script as a template for a new script, copy the file to a new name before making your modifications.

After a condition event occurs, but before the response script executes, ERRM sets a number of environment variables that contain information about the event. The script can check the values of these variables in order to provide the event information to the user. Using the ERRM environment variables, the script can

ascertain such information whether it was triggered by the condition event or rearm event, the time the event occurred, the host on which the event occurred, and so on.

The following example shows the contents of the predefined **wallevent** script for illustration. The **ERRM** environment variables names begin with "**ERRM_**" and are highlighted in the following example.

```
# main()

PERL=/usr/sbin/rsct/perl5/bin/perl

CTMSG=/usr/sbin/rsct/bin/ctdspmsg
MSGMAPPATH=/usr/sbin/rsct/msgmaps
export MSGMAPPATH

Usage=~$CTMSG script IBM.ERrm.cat MSG_SH_USAGE~

while getopts ":h" opt
do
  case $opt in
    h ) print "Usage: `basename $0` [-h] "
        exit 0;;
    ? ) print "Usage: `basename $0` [-h] "
        exit 3;;
    esac
done

# convert time string
seconds=${ERRM_TIME%,*}

EventTime=$(seconds=$seconds $PERL -e \
,
use POSIX qw(strftime);
print strftime("%A %D %T", localtime($ENV{seconds}) );
,
)

WallMsg=~$CTMSG script IBM.ERrm.cat MSG_SH_WALLN "$ERRM_COND_SEVERITY"
"$ERRM_TYPE" "$ERRM_COND_NAME" "$ERRM_RSRC_NAME"
"$ERRM_RSRC_CLASS_NAME" "$EventTime" "$ERRM_NODE_NAME"
"$ERRM_NODE_NAMELIST"~

wall "${WallMsg}"

#wall "$ERRM_COND_SEVERITY $ERRM_TYPE occurred for the condition $ERRM_COND_NAME
on the resource $ERRM_RSRC_NAME of the resource class $ERRM_RSRC_CLASS_NAME at
$EventTime on $ERRM_NODE_NAME"
```

This Perl script uses the **ERRM_TIME** environment variable to ascertain the time that the event occurred, the **ERRM_COND_SEVERITY** environment variable to learn the severity of the event, the **ERRM_TYPE** environment variable to determine if it was the condition event or rearm event that triggered the script's execution, and so on. This information is all included in the message sent to online users. The following table describes the **ERRM** environment variables that you can use in response scripts.

Table 27. Event Response Resource Manager Environment Variables

This environment variable:	Will contain:
ERRM_ATTR_NAME	The display name of the attribute used in the expression that caused this event to occur.

Table 27. Event Response Resource Manager Environment Variables (continued)

This environment variable:	Will contain:
ERRM_ATTR_PNAME	The programmatic name of the attribute used in the expression that caused this event to occur.
ERRM_COND_HANDLE	The resource handle (six hexadecimal integers that are separated by spaces and written as a string) of the condition that caused the event.
ERRM_COND_NAME	The name of the condition that caused the event.
ERRM_COND_SEVERITY	The severity of the condition 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 severity attribute values are represented in this environment variable as a decimal string.
ERRM_COND_SEVERITYID	The severity value of the condition that caused the event. This environment variable will have one of the following values: 0 (Informational), 1 (Warning), or 2 (Critical).
ERRM_DATA_TYPE	The RMC <code>ct_data_type_t</code> of the attribute that changed to cause this event. The following is a list of valid values for this environment variable: <code>CT_INT32</code> , <code>CT_UINT32</code> , <code>CT_INT64</code> , <code>CT_UINT64</code> , <code>CT_FLOAT32</code> , <code>CT_FLOAT64</code> , <code>CT_CHAR_PTR</code> , <code>CT_BINARY_PTR</code> , and <code>CT_SD_PTR</code> . The actual value of the attribute is stored in the ERRM_VALUE environment variable (except for attributes with a data type of <code>CT_NONE</code>).
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 that triggered this event response script.
ERRM_EXPR	The condition event expression or rearm event expression that tested true, thus triggered this linked response. The type of event that triggered the linked response is stored in the ERRM_TYPE environment variable.
ERRM_NODE_NAME	The host name on which this event or rearm event occurred.
ERRM_NODE_NAMELIST	A list of host names. These are the hosts on which the monitored resource resided when the event occurred.
ERRM_RSRC_CLASS_PNAME	The programmatic name of the resource class containing the attribute that changed, thus causing the event to occur.
ERRM_RSRC_CLASS_NAME	The display name of the resource class containing the attribute that changed, thus causing 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 attribute changed, thus causing this event.
ERRM_RSRC_TYPE	The type of resource that caused the event to occur. This environment variable will have one of the following values: 0 (an existing resource), 1 (a new resource), or 2 (a deleted resource).
ERRM_SD_DATA_TYPE	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 <code>CT_SD_PTR</code> . For example: <code>CT_CHAR_PTR</code> , <code>CT_UINT32_ARRAY</code> , <code>CT_UINT32_ARRAY</code> , <code>CT_UINT32_ARRAY</code> .
ERRM_TIME	The time the event occurred. The time is written as a decimal string representing 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 <i>event</i> or <i>rearm event</i> .
ERRM_TYPEID	The value of ERRM_TYPE . This environment variable will have one of the following values: 0 (Event) or 1 (Rearm Event).

Table 27. Event Response Resource Manager Environment Variables (continued)

This environment variable:	Will contain:
ERRM_VALUE	<p>The value of the attribute that caused the event to occur for all attributes except those with a data type of CT_NONE.</p> <p>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.</p> <p>CT_CHAR_PTR is represented as a string for this environment variable.</p> <p>CT_BINARY_PTR is represented as a hexadecimal string separated by spaces.</p> <p>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.</p>
Note:	<p>In addition to these ERRM environment variables, you can, when defining a response action using either the mkresponse or chresponse command, specify additional environment variables for RMC to set prior to triggering the event response script. This enables you to write a more general purpose script that will behave differently based on the environment variables settings associated with the action. To specify such user-defined environment variables, use the -E flag of either the mkresponse or chresponse command. For example:</p> <pre>mkresponse -n "Page Admins" -s /usr/sbin/rsct/bin/pageevent -d 1+7 -t 0000-2400 -e a -E 'ENV1="PAGE ALL"' "contact system administrators"</pre>

Of course, if you do create your own response scripts, you should test them before using them as actions in a production environment. The **-o** flag of the **mkresponse** and **chresponse** commands is useful when debugging new actions. When specified, all standard output from the script is directed to the audit log. This is useful because, while standard error is always directed to the audit log, standard output is not.

For more information about the predefined response scripts (as well as information on the **-E** and **-o** flags of the **mkresponse** and **chresponse** commands), refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Modifying a response

To modify a response, you use the **chresponse** command. You can use this command to:

- add actions to the response
- remove actions from the response
- rename the response

For adding an action, the **chresponse** command uses the same flags as the **mkresponse** command. You specify the **-a** flag to indicate that you want to add an action, and then define the action using the flags described in “Creating a response” on page 114. For example, the following command adds an action to a response named “log info”.

```
chresponse -a -n LogAction -s /usr/sbin/rsct/bin/logevent /tmp/EventLog
-d 1+7,2-6 -t 0000-2400,0800-1700 -e a "log info"
```

To delete an action from a response specify the **-p** flag on the **chresponse** command. You’ll also need to specify the action you want to remove using the **-n** flag. To remove the action named “E-mail root” from the response named “E-mail root any time”, you would enter the following command:

```
chresponse -p -n "E-mail root" "E-mail root any time"
```

To rename a response, you use the **-c** flag. For example, to rename the response "E-mail root any time" to "E-mail system administrator", you would enter:

```
chresponse -c "E-mail system administrator" "E-mail root any time"
```

If the response you want to modify is defined on another node of a peer domain or management domain, you can specify the node name along with the response. For example:

```
chresponse -a -n LogAction -s /usr/sbin/rsct/bin/logevent /tmp/EventLog  
-d 1+7,2-6 -t 0000-2400,0800-1700 -e a "log info":nodeA
```

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the CT_MANAGEMENT_SCOPE environment variable) for the local node or the node specified by the CT_CONTACT environment variable (if it is set). For more information, refer to the **chresponse** command man page and “How do I determine the target nodes for a command?” on page 75.

You will not be able to modify a response that is locked. Instead, the **chresponse** command will generate an error informing you that the response is locked. For more information on unlocking a response so it can be modified, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

For detailed syntax information on the **chresponse** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Removing a response

The **rmresponse** command enables you to remove a response. For example:

```
rmresponse "E-mail system administrator"
```

If the response you have specified has linked conditions, an error message will display and the response will not be removed. To remove the response even if it has linked conditions, use the **-f** (force) flag. For example:

```
rmresponse -f "E-mail system administrator"
```

If the response you want to remove is defined on another node of a peer domain or management domain, you can specify the node name along with the response. For example:

```
rmresponse "E-mail system administrator":nodeA
```

You will not be able to remove a response that is locked. Instead, the **rmresponse** command will generate an error informing you that the response is locked. For more information on unlocking a response so it can be removed, refer to “Locking and unlocking conditions, responses, and condition/response links” on page 128.

Targeting Node(s):

When specifying a node as in the preceding example, the node specified must be a node defined within the management scope (as determined by the CT_MANAGEMENT_SCOPE environment variable) for the local node or the node specified by the CT_CONTACT environment variable (if it is set). For more information, refer to the **chresponse** command man page and “How do I determine the target nodes for a command?” on page 75.

For detailed syntax information on the **rmresponse** command, refer to its online man page. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Querying CIM properties

The Common Information Model (CIM) is a data model for organizing computer hardware and software resources into a common object-oriented class hierarchy. Developed and maintained by the Distributed Management Task Force (DMTF), CIM is a conceptual model for extracting management information. In this way, it is similar to the RMC data model.

The CIM resource manager is an RMC resource manager that enables you to use RMC to query system configuration through CIM classes. The CIM resource manager provides a command (**mkcimreg**) that enables you to register CIM classes with RMC. Once registered, you can query the value of CIM properties using the RMC command **lsrsrc**. This section describes how to query CIM properties through RMC, but does not describe the CIM standard in detail. For complete information on the CIM standard, refer to DMTF's web page at:

<http://www.dmtf.org>

As already stated, CIM is conceptually similar to the RMC data model. Before describing how to query CIM properties through RMC, it is useful to understand the key terminology differences between the CIM and RMC data models. These differences are outlined in the following table.

This CIM term:	Is analogous to the RMC term:	These terms refer to:
Provider	Resource Manager	Processes that can set or return information about a physical or software entity. Defines a number of resource classes (<i>classes</i> in CIM terminology).
Class	Resource Class	The set of resources (<i>instances</i> in CIM terminology) of a common type.
Instance	Resource	The logical abstractions that represent the actual physical or software resources (<i>managed objects</i> in CIM terminology).
Property	Attribute	These terms refer to a characteristic of a resource (<i>instance</i> in CIM terminology).
Managed Object	Physical or Software Resource	The actual hardware or software entity that is represented as a resource (<i>instance</i> in CIM terminology) by a particular resource manager (<i>provider</i> in CIM terminology).

To monitor a CIM property through RMC, you first need to register the appropriate CIM class and Common Manageability Programming Interface (CMPI) provider with RMC. CIM RM supports only 32-bit CMPI providers.

Special Requirement for CSM Users

If you are using the Cluster Systems Management (CSM) product, please note that any CIM class and provider registered on a managed node that needs to be queried from the management server must also be installed on the management server. Because the CIM resource manager is available only in Linux RSCT installations, this requirement also means that the management server must be running a supported Linux distribution if you want to query registered CIM classes on managed nodes.

To register a CIM class and CMPI provider, use the CIM resource manager's **mkcimreg** command. You supply the **mkcimreg** command with a list of Managed Object Format (MOF) files containing either CIM class definitions or provider registration information. The command then outputs files used by the CIM resource manager to enable RMC to work with the CIM classes.

For a current list of CIM classes supported by the CIM resource manager, refer to the "CIM Classes" section of the read-only file `/usr/sbin/rsct/cfg/ct_class_ids`. The CIM classes begin with the characters *cimv2*. Currently, only CIM classes listed in `ct_class_ids` that are also instrumented by Instance providers will yield useful data.

On Linux Nodes:	On AIX Nodes:
<p>The class and provider MOF files and the provider library files for most of the classes listed are available from the Standards Based Linux Instrumentation for Manageability (SBLIM) web site. SBLIM is an IBM Open Source project whose web site is located at:</p> <p>http://www-124.ibm.com/sblim/instrumentation.html</p> <p>The SBLIM providers are also available as part of the SUSE Linux Enterprise Server 9 (SLES 9) distribution. To access the providers, the <code>sblim-cmpi</code> packages must be installed. Once installed, the provider MOF files will be available in the <code>/usr/pegasus/provider/mof</code> directory.</p>	<p>The class and provider MOF files and the provider library files for the AIX classes listed in the <code>ct_class_ids</code> file are provided with the Pegasus CIM Server and OS base providers package. The Pegasus CIM Server and base providers are part of the AIX 5L Expansion Pack. To use the CIM classes provided by the OS base providers, install the following packages off the expansion pack:</p> <p>sysmgt.pegasus.cimserver installs the Pegasus CIM Server filesets in the <code>/opt/freeware/cimom/pegasus</code> directory.</p> <p>sysmgt.pegasus.osbaseproviders installs the base providers for AIX filesets in the <code>/usr/pegasus/provider</code> directory.</p> <p>For more information on the Pegasus CIM Server, refer to the <i>AIX 5L: Common Information Model Guide</i>, SC23-4942.</p>

In order to query one of the CIM classes listed in `/usr/sbin/rsct/cfg/ct_class_ids`, you will need to register both the CIM class and CIM provider using the **mkcimreg** command. The appropriate class and provider MOF files must also be available on your file system. To register CIM classes and providers:

1. Shut down the CIM resource manager using the **stopsrc** command. Use the **stopsrc** command's **-s** flag to identify the CIM resource manager (*IBM.CIMRM*).

```
stopsrc -s IBM.CIMRM
```

2. Make sure CIM resource manager has shut down by issuing the **lssrc** command. Use the **lssrc** command's **-s** flag to indicate that you want the status of the CIM resource manager (*IBM.CIMRM*).

```
lssrc -s IBM.CIMRM
```

Output will be similar to the following. Make sure that the output shows the status of the CIM resource manager to be inoperative. If it is not inoperative, repeat this step.

```
Subsystem      Group          PID    Status
IBM.CIMRM      rsct_rm        6261   inoperative
```

3. Register one or more CIM classes by supplying the **mkcimreg** command with the path(s) to the MOF file(s).

Note: You cannot register classes that derive from classes that have not yet been registered. When you have a class derived from another, be sure to register the parent class first.

To register the CIM classes in the MOF file `Linux_Base.mof` located in the current directory, you would enter:

```
mkcimreg Linux_Base.mof
```

To register the CIM classes in the MOF files *Linux_Base.mof* and *Linux_Network.mof*, you would enter:

```
mkcimreg Linux_Base.mof Linux_Network.mof
```

You can also use the **-I** flag on the **mkcimreg** command to specify one or more additional directories to be searched for MOF files. For example, if the MOF files are all located in */u/jbrady/MOF*, you could enter:

```
mkcimreg -I /u/jbrady/MOF Linux_Base.mof Linux_Network.mof
```

If a class specified on the **mkcimreg** command has already been registered, the **mkcimreg** command will not register the class again and will instead return an error. If you are attempting to register a new version of the class, you can use the **-f** flag to force the class to be registered again.

For example:

```
mkcimreg -f Linux_Base.mof
```

When registering a new version of the class using the **-f** flag, you must also register all subclasses of the upgraded class in order to propagate the changes introduced in the new class to its subclasses. Since the changes propagate from parent class to child class, you must reregister the entire class hierarchy in descending order starting with the topmost parent class and finishing with the lowest-level child class.

4. Register the CIM provider(s) by supplying the **mkcimreg** command with the path to the directory containing the provider library files and the path(s) to the provider MOF file(s). You specify the provider library file directory using the **-p** flag. For example, the provider MOF files associated with the *Linux_Base.mof* and *Linux_Network.mof* files are *Linux_BaseRegistration.mof* and *Linux_NetworkRegistration.mof*. If the library files for these providers were located in */usr/lib* and the MOF files were in the current directory, you could register them by entering:

```
mkcimreg -p /usr/lib Linux_BaseRegistration.mof Linux_NetworkRegistration.mof
```

5. The **mkcimreg** command outputs a number of files which describe new RMC resource classes for the CIM classes defined in the MOF files. In order to detect this new resource class information, you will need to stop the CIM resource manager, and stop and restart the RMC subsystem.

Attention: The **rmcctr -k** command described in the following procedure shuts down RMC. Any RMC-dependent resource monitoring in place at the time is deactivated. Environments relying on RMC or any of its resource managers for high availability or other critical system functions may become temporarily disabled.

- a. To stop the RMC subsystem, issue the **rmcctrl** command with its **-k** flag.

```
rmcctrl -k
```

The **rmcctrl** command is located in */usr/sbin/rsct/bin*. Add this directory to your PATH, or specify the full path on the command line.

- b. Make sure RMC subsystem has shut down by issuing the **lsrsrc** command. Use the **lsrsrc** command's **-s** flag to indicate that you want the status of the RMC subsystem (*ctrmc*).

```
lsrsrc -s ctrmc
```

Output will be similar to the following. Make sure that the output shows the status of the RMC subsystem to be inoperative. If it is not inoperative, repeat this step.

Subsystem	Group	PID	Status
ctrmc	rsct	6199	inoperative

- c. To restart the RMC subsystem, issue the **rmcctrl** command with its **-A** flag.

```
rmcctrl -A
```

When you registered the CIM class and its provider, the CIM classes defined in the MOF files were mapped to new RMC resource classes. The RMC resource class name will be a concatenation of the namespace and the CIM class name — for example *cimv2.Linux_ComputerSystem*. All registered CIM classes are placed in the *root/cimv2* namespace.

Now that you have restarted the RMC subsystem, RMC will have detected these new classes. To verify that the resource classes were created, issue the **lsrsrc** command without any options to list all resource classes.

```
lsrsrc
```

Output will be similar to the following. The resource classes created for the CIM classes defined in *Linux_Base.mof* and *Linux_Network.mof* are highlighted in this example.

```
class_name
"IBM.Association"
"IBM.AuditLog"
"IBM.AuditLogTemplate"
"IBM.Condition"
"IBM.EthernetDevice"
"IBM.EventResponse"
"IBM.Host"
"IBM.FileSystem"
"IBM.Program"
"IBM.TokenRingDevice"
"IBM.Sensor"
"IBM.PeerDomain"
"IBM.PeerNode"
"IBM.RSCTParameters"
"IBM.NetworkInterface"
"IBM.CommunicationGroup"
"IBM.HostPublic"
"IBM.TieBreaker"
"cimv2.Linux_ComputerSystem"
"cimv2.Linux_OperatingSystem"
"cimv2.Linux_UnixProcess"
"cimv2.Linux_Processor"
"cimv2.Linux_RunningOS"
"cimv2.Linux_OSProcess"
"cimv2.Linux_CSProcessor"
"cimv2.Linux_IPProtocolEndpoint"
"cimv2.Linux_LocalLoopbackPort"
"cimv2.Linux_EthernetPort"
"cimv2.Linux_TokenRingPort"
"cimv2.Linux_CSNetworkPort"
"cimv2.Linux_NetworkPortImplementsIPEndpoint"
```

You can query the properties of any of these new resource classes in the same way you would query any property in RMC. You will only see actual resources for a class if it has a CMPI Instance provider registered that supports the class. Issue the **lsrsrc** command, supplying it with the resource class name as an argument. For example, to list the properties for the *cim2.Linux_ComputerSystem* resource class, enter:

```
lsrsrc cimv2.Linux_ComputerSystem
```

Output will be similar to the following:

```
Resource Persistent Attributes for: cimv2.Linux_ComputerSystem
resource 1:
    NameFormat          = "IP"
```

```

Dedicated          = {0}
CreationClassName  = "Linux_ComputerSystem"
Name               = "c175nf01.ppd.pok.ibm.com"
PrimaryOwnerName   = "root"
PrimaryOwnerContact = "root@c175nf01.ppd.pok.ibm.com"
EnabledState       = 2
OtherEnabledState  = "NULL"
RequestedState     = 2
EnabledDefault     = 2
Status             = "NULL"
Caption            = "Computer System"
Description         = "A class derived from ComputerSystem that represents
the single node container of the Linux OS."
ElementName        = "c175nf01.ppd.pok.ibm.com"
ActivePeerDomain   = ""

```

For detailed attribute definition information, use the **lsrsrdef** command. For example:

```
lsrsrdef -e cimv2.Linux_ComputerSystem
```

Returns detailed attribute information for the *cimv2.Linux_ComputerSystem* resource class.

Resource Persistent Attribute Definitions for: cimv2.Linux_ComputerSystem
attribute 1:

```

program_name = "NameFormat"
display_name = "NameFormat"
group_name   = "description is not available"
properties   = {"option_for_define","selectable","public"}
description  = "The ComputerSystem object and its derivatives are Top Level Objects of CIM. They provide the scope for numerous components. Having unique System keys is required. The NameFormat property identifies how the ComputerSystem Name is generated. The NameFormat ValueMap qualifier defines the various mechanisms for assigning the name. Note that another name can be assigned and used for the ComputerSystem that better suit a business, using the inherited ElementName property."
attribute_id = 0
group_id     = 0
data_type    = "char_ptr"
variety_list = {[1,1]}
variety_count = 1
default_value = ""

```

attribute 2:

```

program_name = "OtherIdentifyingInfo"
display_name = "OtherIdentifyingInfo"
group_name   = "description is not available"
properties   = {"option_for_define","selectable","public"}
description  = "OtherIdentifyingInfo captures additional data, beyond System Name information, that could be used to identify a ComputerSystem. One example would be to hold the Fibre Channel World-Wide Name (WWN) of a node. Note that if only the Fibre Channel name is available and is unique (able to be used as the System key), then this property would be NULL and the WWN would become the System key, its data placed in the Name property."
attribute_id = 1
group_id     = 0
data_type    = "char_ptr_array"
variety_list = {[1,1]}
variety_count = 1
default_value = {""}

```

attribute 3:

```
program_name = "IdentifyingDescriptions"
```

```

.
.
.

```

For detailed syntax information on the **mkcimreg**, **rmcctrl**, **lsrsrc**, and **lsrsrcdef** commands, refer their online man pages. Detailed syntax information is also provided in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Catching SNMP traps on Linux nodes

Note: The ability to catch SNMP trap messages described in this section is available on Linux nodes only. This capability is not available as part of the AIX implementation of RSCT.

The Simple Network Management Protocol (SNMP), a standard operations and maintenance protocol, uses trap-directed notification for receiving information about managed devices. Instead of polling each managed device, which can be resource intensive, an agent on a managed device can send unsolicited messages when events of interest occur. These unsolicited messages are known as SNMP “traps”.

If you have an SNMP-managed network, you can use RMC on Linux nodes to catch SNMP traps. You can use RMC’s event management capabilities to respond to the trap message as you would respond to a monitored event in RMC. The SNMP trap information is also entered into the audit log. To catch SNMP traps:

1. Run the **cfgrmcsmnp** command. This command will configure the node to receive SNMP traps.

```
cfgrmcsmnp
```

The **cfgrmcsmnp** command is located in **/usr/sbin/rsct/install/bin**. Add this directory to your PATH, or specify the full path on the command line.

When a node is configured to receive SNMP traps, a sensor object named **SNMPTrap** is added to the RMC subsystem. When an SNMP trap is received, the String dynamic attribute of the **SNMPTrap** sensor object will be updated to reflect the trap information. The String dynamic attribute will contain the trap origin, type, and value information separated by newline characters. For example, issuing the following command to generate a trap:

```
snmptrap -v 2c -c public localhost '' 0 0 s "Hello, this is an SNMP trap."
```

would cause the String attribute of the **SNMPTrap** sensor to be updated. Using the generic RMC command **lsrsrc**, you can display the trap information. The command:

```
lsrsrc -s "Name='SNMPTrap'" IBM.Sensor String
```

Would return:

```
Resource Persistent Attributes for IBM.Sensor  
resource 1:
```

```
String = SNMP Trap from localhost.localdomain (127.0.0.1)\nTrap Ty  
pe: zeroDotZero\nOID: zeroDotZero VALUE: Hello, this is an SNMP trap.
```

2. A predefined condition named “SNMP trap detected” will have been created when RSCT was installed. Use the **mkcondresp** command to associate this condition with a response of your choice. You can use one of the predefined responses, or you can create one of your own as described in “Creating a response” on page 114.

The following example associates the “SNMP trap detected” condition with the predefined response “Broadcast details of event any time”.

```
mkcondresp "SNMP trap detected" "Broadcast details of event any time"
```

3. Start condition monitoring (SNMP trap detection) using the **startcondresp** command:

```
startcondresp "SNMP trap detected"
```

To verify that the condition is being monitored, you can use the **lscondition** command:

```
lscondition
```

Output is similar to:

```
Displaying condition information:
Name           MonitorStatus
"SNMP trap detected" "Monitored"
```

- To later stop SNMP trap detection, you can use the **stopcondresp** command:

```
stopcondresp "SNMP trap detected"
```

- To verify that the condition is no longer being monitored, you can use the **lscondition** command:

```
lscondition
```

Output is similar to:

```
Displaying condition information:
Name           MonitorStatus
"SNMP trap detected" "Not monitored"
```

- To unconfigure the ability to detect SNMP traps on the node, enter the **cfgmcsnmp** command with its **-u** flag:

```
cfgmcsnmp -u
```

For detailed syntax information on the **cfgmcsnmp**, **mkcondresp**, **startcondresp**, and **stopcondresp** commands, refer to their online man pages. Detailed syntax information is also available in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Locking and unlocking conditions, responses, and condition/response links

Conditions, responses, and condition/response links can be locked to prevent user modification or removal. A locked condition, response, or condition/response link cannot be modified or removed until it is unlocked. For this reason, the following commands for manipulating conditions, responses, and condition/response links may fail to make the expected change if the resource you are trying to manipulate with the command is locked. Instead, an error will be generated informing you that the condition, response, or condition/response link is locked. The commands that will fail to act upon a particular locked resource are:

- the **chcondition** command which modifies a condition. A locked condition cannot be modified.
- the **chresponse** command which modifies a response. A locked response cannot be modified.
- the **rmcondition** command which removes a condition. A locked condition cannot be removed.
- the **rmcondresp** command which deletes the link between a condition and response. A locked condition/response link cannot be removed.

- the **rmresponse** command which removes a response. A locked response cannot be removed.
- the **startcondresp** command which starts monitoring a condition that has one or more linked responses. If the condition/response link is locked, you will not be able to start monitoring.
- the **stopcondresp** command which stops monitoring a condition that has one or more linked responses. If the condition/response link is locked, you will not be able to stop monitoring.

System software that uses RSCT may lock certain monitoring resources that are considered vital for the system software to work properly. Similarly, as a system administrator, you may choose to lock certain monitoring resources that you consider vital in order to prevent accidental modification or removal.

Two flags (**-L** and **-U**) are provided on a number of Event Response Resource Manager commands to enable you to lock and unlock monitoring resources. The **-L** flag locks the condition, response, or condition/response link, while the **-U** flag unlocks it.

Before using the **-U** flag as described in this section, you should be aware that if a particular condition, response, or condition/response link has been locked, this may be because it is essential for system software to work properly. For this reason, you should exercise caution before unlocking a condition, response, or condition/response link. In general, if you do not know why the monitoring resource is locked, you should not unlock it.

Locking or unlocking a condition

To lock or unlock a condition, use the **-L** and **-U** flags on the **chcondition** command. When using either of these flags, no other operation can be performed by the **chcondition** command. The syntax is:

```
chcondition {-L | -U} condition[:node_name]
```

For example, if you have created a condition named `/usr space used` and now want to lock it to prevent accidental modification or removal, you would enter:

```
chcondition -L "/usr space used"
```

To unlock this condition, you would enter:

```
chcondition -U "/usr space used"
```

Locking or unlocking a response

To lock or unlock a response, use the **-L** and **-U** flags on the **chresponse** command. When using either of these flags, no other operation can be performed by the **chresponse** command. The syntax is:

```
chresponse {-L | -U} response[:node_name]
```

For example, if you have created a response named `log info to /tmp/EventLog` and now want to lock it to prevent accidental modification or removal, you would enter:

```
chresponse -L "log info to /tmp/EventLog"
```

To unlock this response, you would enter:

```
chresponse -U "log info to /tmp/EventLog"
```

Locking or unlocking a condition/response link

To lock or unlock a condition/response link, use the **-L** and **-U** flags on either the **rmcondresp** command, the **startcondresp** command, or the **stopcondresp** command. The **-L** and **-U** flags perform the exact same operation regardless of which of the commands you use. No other operation can be performed by these commands when you use the **-L** or **-U** flag.

The syntax for locking or unlocking a condition/response link using the **rmcondresp** command is:

```
rmcondresp {-L | -U} condition[:node_name] response
```

The syntax for locking or unlocking a condition/response link using the **startcondresp** command is:

```
startcondresp {-L | -U} condition[:node_name] response
```

The syntax for locking or unlocking a condition/response link using the **stopcondresp** command is:

```
stopcondresp {-L | -U} condition[:node_name] response
```

For example, say you have created a link between a condition /usr space used and a response log info to /tmp/EventLog and started monitoring. To prevent a user from accidentally stopping monitoring, you could lock this condition/response link. Since the **-L** flag is provided on the **rmcondresp** command, the **startcondresp** command, and the **stopcondresp** command, any of the following commands will lock the condition/response link.

```
rmcondresp -L "/usr space used" "log info to /tmp/EventLog"
```

```
startcondresp -L "/usr space used" "log info to /tmp/EventLog"
```

```
stopcondresp -L "/usr space used" "log info to /tmp/EventLog"
```

Similarly, any of the following commands will unlock the condition/response link so it can be stopped, started, or removed.

```
rmcondresp -U "/usr space used" "log info to /tmp/EventLog"
```

```
startcondresp -U "/usr space used" "log info to /tmp/EventLog"
```

```
stopcondresp -U "/usr space used" "log info to /tmp/EventLog"
```

Using expressions to specify condition events and command selection strings

An expression in RMC 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. This section provides more detailed information (such as permissible data types, operators, and operator precedence) about expressions.

There are two types of expressions you can specify on certain RMC and ERRM commands described throughout this chapter. One type is the event expression/rearm event expressions you define for conditions using the

mkcondition or **chcondition** command. Event expressions are described in “What is an event expression?” on page 70 and “What is a rearm event expression?” on page 71.

The other type of expression you can specify on certain RMC and ERRM commands is a *selection string expression*. A number of commands described in this chapter enable you to specify a selection string expression that restricts the command action in some way. The commands that accept a selection string expression are summarized in the following table. For general information about how the selection strings are used by these commands, refer to the sections referenced in the table. You can also find complete syntax information on any of these commands in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Table 28. Commands whose actions you can restrict using selection strings

This command:	Does This:	The Command’s Selection String Expression:	For more information on this command, refer to:
chcondition	Changes the attributes of a condition. The condition monitors an attribute of one or more resources of a specified class.	Restricts the command to a subset of the resources in the resource class. The selection string expression filters the available resources by one or more persistent attributes of the resource class. The defined condition will monitor the attribute for only those resources that match the selection string.	“Modifying a condition” on page 111.
chsrc	Changes persistent attribute values of a resource within a specified resource class.	Identifies the resource within the resource class. The selection string expression filters the available resources by one or more persistent attributes of the resource class.	<i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i>
lsaudrec	Lists records from the audit log.	Filters the audit log so that only records that match the selection string are listed. The selection string expression filters the audit log using one or more record field names.	“Using the audit log to track monitoring activity” on page 93.
lsrsrc	Lists resources of a resource class.	Restricts the command to a subset of the resources in the resource class. The selection string expression filters the available resources by one or more persistent attributes of the resource class. Only the resource(s) that match the selection string will be listed.	<i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i>
mkcondition	Creates a new condition. The condition monitors an attribute of one or more resources of a specified class.	Restricts the command to a subset of the resources in the resource class. The selection string expression filters the available resources by one or more persistent attributes of the resource class. The defined condition will monitor the attribute for only those resources that match the selection string.	“Creating a condition” on page 101.
rmaudrec	Removes records from the audit log.	Specifies the set of records in the audit log that should be removed. The selection string identifies the records using one or more record field names. Only records that match the selection string are removed.	“Deleting entries from the audit log” on page 96.

Table 28. Commands whose actions you can restrict using selection strings (continued)

This command:	Does This:	The Command's Selection String Expression:	For more information on this command, refer to:
rmrsrc	Removes resources of a specified resource class.	Restricts the command to a subset of the resources in the resource class. The selection string expression filters the available resources by one or more persistent attributes of the resource class. Only the resource(s) that match the selection string will be removed.	<i>Reliable Scalable Cluster Technology for AIX 5L: Technical Reference</i> and the <i>Reliable Scalable Cluster Technology for Linux: Technical Reference</i> .

SQL restrictions

SQL syntax is supported for selection strings. The following table relates the RMC terminology to SQL terminology.

Table 29. Relationship of RMC terminology to SQL terminology

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

Although SQL syntax is generally supported in selection strings, the following restrictions apply.

- 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:

Table 30. Supported Base Data Types

Symbolic Name	Description
CT_INT32	Signed 32-bit integer
CT_UINT32	Unsigned 32-bit integer
CT_INT64	Signed 64-bit integer

Table 30. Supported Base Data Types (continued)

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 133, 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 usually 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 RMC's previous observation of the attribute value. Because RMC polls attribute values periodically and keeps track of the previously observed value, you can use this syntax to compare the currently observed value with the previously observed value. For example, the following event expression would trigger an event if the average number of processes on the run queue has increased by 50% or more between observations:

```
(ProcRunQueue - ProcRunQueue@P) >= (ProcRunQueue@P * 0.5)
```

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:

Table 31. Operators That Can Be Used in Expressions

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	"0x0fff<<4" results in 0xffff0	None
==	Equality	All but SDs	All but SDs	"2==2" results in 1 "2=2" results in 1	Result is true (1) or false (0)
!=	Inequality	All but SDs	All but SDs	"2!=2" results in 0	Result is true (1) or false (0)
<>				"2<>2" results in 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. To use this operator in an expression, the locale(s) of the node(s) running the RMC daemon must be using either Unicode Transfer Format-8 (UTF-8) encoding (or a codeset that matches UTF-8), or else C locale encoding. If multiple nodes are involved, the encoding must be consistent across all nodes.

Table 31. Operators That Can Be Used in Expressions (continued)

Operator	Description	Left Data Types	Right Data Types	Example	Notes
!~	Not pattern match	Strings	Strings	"abc"!~"a.*" results in 0	Right operand is interpreted as an extended regular expression. To use this operator in an expression, the locale(s) of the node(s) running the RMC daemon must be using either Unicode Transfer Format-8 (UTF-8) encoding (or a codeset that matches UTF-8), or else C locale encoding. If multiple nodes are involved, the encoding must be consistent across all nodes.
=? 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
 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.

Table 32. Operator Precedence

Operators	Description
.	Structured data element separator
~	Bitwise complement
!	Logical not
NOT	
not	
-	Unary minus
+	Unary plus
*	Multiplication
/	Division
%	Modulo
+	Addition
-	Subtraction
<<	Left shift
>>	Right shift
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal

Table 32. Operator Precedence (continued)

Operators	Description
==	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 5. Controlling access to root commands and scripts

The RSCT least-privilege (LP) resource manager is a client-server application that allows you to enhance the security, performance, and control of applications that require root authority to run. The LP resource manager runs on both AIX and Linux nodes. Through the LP resource manager, you can:

- Define specific root commands or scripts as LP resources. An LP resource represents a least-privilege access command or script. Least-privilege capability allows a select group of authorized users to run the command or script without needing complete root authority.
- Enable distributed and parallel execution of these LP resources. Authorized users can run the command or script locally or remotely, on one or simultaneously on many nodes, without having to log into each node on which the command or script is to run.
- Monitor and manage LP resources and operations on one node or across many nodes. The LP resource manager uses the Audit log resource manager to log detailed usage information about LP resources.

Use the following roadmap of topics to learn more about using the LP resource manager. Many of these topics and instructions refer to RSCT commands and to the IBM.LPCommand resource class. For complete descriptions, refer to the *RSCT for AIX 5L: Technical Reference* or the *RSCT for Linux: Technical Reference*. The same reference information can be found for any command by viewing its online man page.

Subtask	Associated information or instructions (see . . .)
Learn about the LP resource manager, its associated resource class and commands	"Overview of LP resource manager operation"
Determine the target nodes of an LPRM operation before you issue an LPRM command	"Determining the target nodes for an LPRM command" on page 142
Monitor LP resources and operations	"Monitoring LP resources and operations" on page 143
Define LP resources and authorized users	"Steps for defining LP resources and authorized users" on page 143
Use and manage LP resources	<ul style="list-style-type: none">• "Step for running an LP resource" on page 144• "Steps for changing an LP resource" on page 145• "Steps for removing LP resources" on page 145

Overview of LP resource manager operation

The LP resource manager consists of two parts, a client program and a daemon. Instances of both the client and daemon run on each node, on AIX or Linux. The nodes may be independent workstations, or may be in a management or peer domain.

The LP resource manager provides one resource class, IBM.LPCommands, that represents root commands or scripts. Through this representation of resources, the

LP resource manager can run a root command or script, locally or remotely, on behalf of an authorized user. When the resource's processing completes, the LP resource manager returns the processing results to the user. More specifically, the resource manager:

- Allows administrators to manage LP resources by defining, changing, and removing them. Administrators may use not only resource monitoring and control (RMC) commands to manage LP resources, but also the following LPRM commands.

chlpcmd	Changes certain attributes of an LP resource.
lphistory	Lists a particular number of previously issued LPRM commands.
lslpcmd	Lists one or more LP resources on one or more nodes in a domain.
mklpcmd	Defines an LP resource to the RMC subsystem.
rmlpcmd	Removes an LP resource from one or more nodes in a domain.
runlpcmd	Runs a particular LP resource on one or more nodes in a domain.
- Enables local or remote execution of the LP resources from one or more nodes within a management or peer domain. Two environment variables, CT_CONTACT and CT_MANAGEMENT_SCOPE, affect which LPRM daemon runs and its scope of operation. Further details appear in “Determining the target nodes for an LPRM command.”
- Secures access to the root commands or scripts by using cluster technology security services to authenticate, and the RMC subsystem's access control list (ACL) to authorize users. For more information about the ACL file, see “Managing user access to resources using RMC ACL files” on page 77.

Determining the target nodes for an LPRM command

You can run LPRM commands on a single machine, on all the nodes of a peer domain, or on all the nodes of a management domain. The LPRM commands enable you to refine this capability even further, allowing you to specify a subset of nodes in the peer domain or management domain. Two environment variables that, together with various command flags, determine the nodes that will be affected by the LPRM commands you enter:

CT_CONTACT

Determines the system that is used for the session with the RMC daemon. When the CT_CONTACT environment variable is set to a host name or IP address, the command contacts the RMC daemon on the specified host. If the environment variable is not set, the command contacts the RMC daemon on the local system where the command is being run. The target of the RMC daemon session and the management scope determine the LP resources that are processed.

CT_MANAGEMENT_SCOPE

Determines the management scope that is used for the session with the RMC daemon to process the LP resources. The management scope determines the set of possible target nodes where the resources can be processed. The valid values are:

- 0** Specifies *local* scope.
- 1** Specifies *local* scope.
- 2** Specifies *peer domain* scope.
- 3** Specifies *management domain* scope.

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

Monitoring LP resources and operations

The LP resource manager provides two commands for monitoring LP resources and operations on one node or across many nodes:

lslpcmd

This command returns a list of the root commands or scripts that are defined as LP resources. Depending on the parameters and flags that you specify, the list contains either the names of LP resources, or the names plus attributes of LP resources. To use this LPRM command, you need to have read permission to the IBM.LPCCommands resource class.

lphistory

This command lists the LPRM commands that were issued since the LP resource manager was started. Through the *number_of_commands* parameter, you may specify the number of commands that you want returned in the list. To use this LPRM command, you need to have write permission to the IBM.LPCCommands resource class.

Further details about these commands appear in the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

In addition, you may use the dynamic attributes of the IBM.LPCCommands resource class to create your own conditions for monitoring. For more information, see “Advanced resource monitoring” on page 98.

The LP resource manager also uses the Audit log resource manager to log detailed usage information about LP resources. For more information about the Audit log resource manager, see the following topics:

- “What resource managers are provided with RSCT?” on page 65
- “Using the audit log to track monitoring activity” on page 93

Steps for defining LP resources and authorized users

Use the **mklpcmd** command to create an LP resource, and the RMC ACL file to authorize users.

Before you begin:

- You need to have write permission to the IBM.LPCCommands resource class on all the nodes where the LP resource will be created. Follow the appropriate instructions in “Managing user access to resources using RMC ACL files” on page 77 to modify the RMC ACL file.
- You need to determine what values to set for the CT_CONTACT and CT_MANAGEMENT_SCOPE environment variables on the **mklpcmd** command. To do so, use the information in “Determining the target nodes for an LPRM command” on page 142.

Perform the following steps to define an LP resource and its authorized users.

1. Determine which users require access to this LP resource. Then follow the appropriate instructions in “Managing user access to resources using RMC ACL files” on page 77 to modify the RMC ACL file to add user identifiers to the stanza for the IBM.LPCCommands resource class.

2. Determine the location where the root command or script will reside. You will need the fully qualified path of the command or script and, optionally, the nodes on which it will be available.
3. Determine whether you want the LP resource manager to validate the command or script, or check for incorrect input, or both, whenever a user issues an LPRM command for the resource you are defining. This decision determines whether you use the default or specify a value for the ControlFlags attribute for the LP resource.
4. Issue the **mklpcmd** command, supplying appropriate values for required parameters and flags.

For example, to define a new LP resource, named *LP1*, pointing to the command */tmp/user1/lpcmd1* on a local node, you would enter:

```
mklpcmd LP1 /tmp/user1/lpcmd1
```

You know you are done when the LP resource manager returns an exit value or message from processing the command or script.

For complete syntax information on the **mklpcmd** command, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*

Step for running an LP resource

Use the **runlpcmd** command to run a root command or script that is defined as an LP resource.

Before you begin:

- You need to have execute permission to the IBM.LPCommand resource class on all the nodes where the LP resource will be run. Follow the appropriate instructions in “Managing user access to resources using RMC ACL files” on page 77 to modify the RMC ACL file.
- You need to determine what values to set for the CT_CONTACT and CT_MANAGEMENT_SCOPE environment variables on the **runlpcmd** command. To do so, use the information in “Determining the target nodes for an LPRM command” on page 142.

Perform the following step to run an LP resource.

1. Issue the **runlpcmd** command, supplying appropriate values for required parameters and flags.

For example, to run the LP resource named *LP1*, which has required input flags and parameters **-a -p User Group**, you would enter:

```
runlpcmd LP1 "-a -p User Group"
```

You know you are done when the LP resource manager returns an exit value or message from processing the command or script.

For complete syntax information on the **runlpcmd** command, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*

Steps for changing an LP resource

Use the **chlpcmd** command to modify an LP resource.

Before you begin:

- You need to have write permission to the IBM.LPCommand resource class on all the nodes where the LP resource will be modified. Follow the appropriate instructions in “Managing user access to resources using RMC ACL files” on page 77 to modify the RMC ACL file.
- You need to determine what values to set for the CT_CONTACT and CT_MANAGEMENT_SCOPE environment variables on the **chlpcmd** command. To do so, use the information in “Determining the target nodes for an LPRM command” on page 142.

Perform the following steps to modify an LP resource.

- (Optional) Use the **lslpcmd** command to display the attribute values for this LP resource. If the resource is locked, you must change the Lock attribute value to 0 before making any additional changes to the resource.
- Issue the **chlpcmd** command, supplying appropriate values for required parameters and flags.

For example, to change the Lock attribute of an LP resource named *LP1*, you would enter:

```
chlpcmd LP1 Lock=0
```

You know you are done when the LP resource manager returns an exit value or message from processing the command or script.

For complete syntax information on the **lslpcmd** and **chlpcmd** commands, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*

Steps for removing LP resources

Use the **rmlpcmd** command to remove an LP resource.

Before you begin:

- You need to have write permission to the IBM.LPCommand resource class on all the nodes where the LP resource will be removed. Follow the appropriate instructions in “Managing user access to resources using RMC ACL files” on page 77 to modify the RMC ACL file.
- You need to determine what values to set for the CT_CONTACT and CT_MANAGEMENT_SCOPE environment variables on the **rmlpcmd** command. To do so, use the information in “Determining the target nodes for an LPRM command” on page 142.

Perform the following steps to remove an LP resource.

1. (Optional) Use the **lslpcmd** command to display the attribute values for this LP resource. If the resource is locked, you must change the Lock attribute value to 0 before attempting to remove the resource.
2. Issue the **rmlpcmd** command, supplying appropriate values for required parameters and flags.

For example, to remove the LP resource named *LP1*, you would enter:

```
rmlpcmd LP1
```

Result: The LP resource manager returns an exit value or message from processing the command or script.

3. (Optional) If necessary, edit the RMC ACL file to remove users from the list for the IBM.LPCommands resource class.

For complete syntax information on the **lsrpcmd** and **rmlpcmd** commands, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* and the *Reliable Scalable Cluster Technology for Linux: Technical Reference*

Chapter 6. Understanding and administering cluster security services

This chapter describes how to administer cluster security services for both an RSCT peer domain and a management domain. For information on creating an RSCT peer domain, refer to Chapter 3, “Creating and administering an RSCT peer domain,” on page 19. For information on creating a management domain, refer to *IBM Cluster Systems Management: Administration Guide*.

RSCT’s cluster security services provides the security infrastructure that enables RSCT components to authenticate and authorize the identity of other parties.

Authentication is the process of ensuring that another party is who it claims to be. Using cluster security services, various cluster applications (such as the configuration resource manager and CSM) can check that other parties are genuine, and not attempting to gain unwarranted access to the system. “Understanding cluster security services’ authentication” describes how authentication is handled on both an RSCT peer domain and a management domain.

Authorization is the process by which a cluster software component grants or denies resources based on certain criteria. Currently, the only RSCT component that implements authorization is RMC, which uses access control list (ACL) files in order to control user access to resource classes and their resource instances. In these ACL files, described in “Managing user access to resources using RMC ACL files” on page 77, you can specify the permissions needed by a user to access particular resource classes and resources. The RMC component subsystem uses cluster security services to map the operating system user identifiers specified in the ACL file with network security identifiers to determine if the user has the correct permissions. This process of mapping operating system user identifiers to network security identifiers is called *native identity mapping*, and is described in “Understanding cluster security services’ authorization” on page 150.

In addition to providing this overview of how authentication and authorization are handled by cluster security services, this chapter will also present a series of administrative tasks you may need or want to perform. Refer to “Cluster security services administration” on page 151 which explains the administrative tasks that are necessary and the steps you need to take to perform them.

Understanding cluster security services’ authentication

Authentication is the process by which a cluster software component, using cluster security services, determines the identity of one of its peers, clients, or an RSCT subcomponent. This determination is made in such a way that the cluster software component can be certain the identity is genuine and not forged by some other party trying to gain unwarranted access to the system. Be aware that authentication is different from authorization (the process of granting or denying resources based on some criteria). Authorization is handled by RMC and is discussed in “Managing user access to resources using RMC ACL files” on page 77.

Cluster Security Services uses **credential based authentication**. This type of authentication is used in client/server relationships and enables:

- a client process to present information that identifies the process in a manner that cannot be imitated to the server.

- the server process to correctly determine the authenticity of the information from the client.

Credential based authentication involves the use of a third party that both the client and the server trust. For this release, only Host Based Authentication is supported, but other security mechanisms may be supported in the future. In the case of Host Based Authentication, the trusted third party is the operating system. This method of authentication is used between RSCT and its client applications (such as CSM).

Understanding credentials based authentication

Credentials based authentication involves the use of a trusted third party to perform authentication in client/server relationships. To enable this type of authentication, cluster security services provides an abstraction layer between the cluster components and the underlying security mechanisms. This abstraction layer is called the Mechanism Abstraction Layer (MAL) and converts mechanism-independent instructions requested by the application into general tasks to be performed by any mechanism. The tasks are carried out by a Mechanism Pluggable Module (MPM). An MPM is a component that converts generalized security services routines into the specific security mechanism functions necessary to carry out a request.

Since Host Based Authentication is currently the only security mechanism supported, there is only one MPM (`/usr/lib/unix.mpm`) available at this time. Additional MPMs to support other security mechanisms may be added in the future. An MPM configuration file is located on each node of your system in the file `/usr/sbin/rsct/cfg/ctsec.cfg`; this file lists the path name of the available MPM. You should **not** modify this file. However, you should be aware that the file exists and is used by cluster security services to locate the MPM.

Understanding Host Based Authentication

Host Based Authentication is the default mechanism provided by cluster security services and utilized by RSCT. This mechanism employs private/public key pairs, associating an unique private/public key pair with each node in the cluster. These keys are used to encrypt and decipher data. Data encrypted with a particular private key can be deciphered only by the corresponding public key, and data encrypted with the public key can be deciphered only by the corresponding private key.

The **ctcsd** daemon provides and authenticates operating system identity based credentials for cluster security services; it is started by the Host Based Authentication MPM. When the cluster security services are installed on a particular node, a private key for the node will be created. From this private key, a public key for the node will be derived. If these keys are not created as part of the installation process, the **ctcsd** daemon will create them when the daemon runs for the first time. The node will use its private key to seal and encrypt data that it transmits. The node's public key will be provided to other nodes and will be used by them to verify and decipher the data. Similarly, the public key can be used by the other nodes to seal and encrypt data that will be verified and deciphered using the node's associated private key. A node's public key is intended to become public knowledge, while the private key remains secret, known only to the node's *root* user.

The private/public key pairs are associated with a node's host name and its active IP addresses. Usually, authentication will be based on a node's host name. For this reason, it is, in most situations, critical that all hosts within the cluster be configured to resolve host names using the same consistent method. If a Domain Name

Service (DNS) is in use, all nodes within the cluster should make use of it. All hosts must be configured to provide host names to applications using either short host names or fully qualified host names (short name plus domain name). If the cluster includes nodes from multiple domains, you **must** use fully qualified host names. If this consistency is not enforced, authentication failures can occur between nodes within the cluster.

Since the public/private key pairs are also associated with a node's known IP address(es), it is, in some cluster configurations and situations, possible to authenticate hosts by their IP addresses, thereby removing the need to resolve host names. Authentication using IP address values alone is possible only in an RSCT peer domain in which all nodes are using version 2.3.1.0 (or later) of RSCT. RSCT versions prior to 2.3.1.0 do not support IP-address authentication. In addition, IP-address authentication is not supported in a CSM management domain.

When the cluster security services are installed on a particular node, the install procedures attempt to create the trusted host list for this node. It is the trusted host list file that associates host identities — host names and IP addresses — with their corresponding public key values. The initial trusted host list file created by the installation process associates the node's public key value with all active host names and IP addresses associated with all configured and active AF_INET and AF_INET6 adapters for this node. If no network interfaces are configured and active at this time, no trusted host list is created by the install procedures. Instead, the **ctcasd** daemon will create this list when it executes for the first time. If a remote node is not listed in a local node's trusted host list, or if the public key recorded for the host is incorrect, the host will not be able to authenticate the node.

Note: If a node's host name or IP address is changed, its private/public key pair does not need to change. You will, however, need to modify the trusted host list file of any node that references the changed node. Specifically, you will need to modify the trusted host list file to include the new host name or IP address, associating it with the existing public key. You should also delete the obsolete host name or IP address from the trusted host list on any node that references it. This is particularly important if the host name or IP address will be reused on another machine. Use the **ctsth1 -d -n {hostname | IP_address}** command to remove obsolete entries.

The following table lists the default locations for a node's private key, public key, and trusted host list.

The default location for a node's:	Is:
private key	<i>/var/ct/cfg/ct_has.qkf</i> This file is readable and accessible only to the root user.
public key	<i>/var/ct/cfg/ct_has.pkf</i> This file is readable to all users on the local system. Write permission is not granted to any system user.
trusted host list	<i>/var/ct/cfg/ct_has.thl</i> This file is readable to all users on the local system. Write permission is not granted to any system user.

Note: You can change the default locations for a node's private key, public key, and trusted host list files by modifying the **ctcasd.cfg** configuration file read

by the **ctcasd** daemon upon startup. If you do choose to change the location of these files, you must modify the **ctcasd.cfg** file as described in “Configuring the **ctcasd** daemon on a node” on page 153. If you do not, the **ctcasd** daemon will not be able to locate the files. This could result in a failure of the **ctcasd** daemon. If the **ctcasd** daemon is unable to locate either the node’s private or public key, it will mistakenly think that it is being started for the first time, and will create a new public/private key pair for the node. These new keys will not match the public keys stored on other cluster nodes, causing authentication failures.

In order for nodes within a cluster to authenticate message signatures during cluster setup, and to create valid Host Based Authentication credentials for RSCT services and their clients, the public keys and their host associations need to be distributed throughout the cluster.

When configuring a cluster of nodes (either as a management domain using CSM or as an RSCT peer domain using configuration resource manager commands), the necessary public key exchanges will, by default, be carried out by CSM or configuration resource manager utilities. If the network is relatively secure against identity and address spoofing, you can use these utilities; if not, the keys should be transferred manually to prevent the inclusion of nodes that are attempting to masquerade as known nodes. You should carefully consider whether the security of the network is sufficient to prevent address and identity spoofing. If you don’t think the network is secure enough, refer to “Guarding against address and identify spoofing when transferring public keys” on page 157. If you are not sure if your network is secure enough, consult with a trained network security specialist to find out if you are at risk.

A node’s private/public key pair are considered synonymous with a node’s identity and are not expected to change over time. However, if a node’s private key does need to be changed, refer to “Changing a node’s private/public key pair” on page 161 for instructions on how to do this.

Understanding cluster security services’ authorization

Authorization is the process by which a cluster software component grants or denies resources based on certain criteria. Currently, the only RSCT component that implements authorization is RMC, which uses access control list (ACL) files in order to control user access to resource classes and their resource instances. In these ACL files, described in “Managing user access to resources using RMC ACL files” on page 77, you can specify the permissions needed by a user to access particular resource classes and resources. The RMC component subsystem uses cluster security services to map the operating system user identifiers specified in the ACL file, with the network security identifiers that are verified by the cluster security services’ authentication process, to determine if the user has the correct permissions. This is called *native identity mapping* and is described next in “Understanding native identity mapping.”

Understanding native identity mapping

This process of mapping operating system user identifiers to network security identifiers is called *native identity mapping*, and is performed by the cluster security services’ *identity mapping service*.

As described in “Understanding credentials based authentication” on page 148, the cluster security services has a Mechanism Abstraction Layer (MAL) that converts

mechanism-independent instructions requested by an application into general tasks to be performed by any mechanism. A Mechanism Pluggable Module (MPM) is a software component that converts generalized security services routines into the specific security mechanism functions necessary to carry out a request. The security context created during authentication is based on the underlying security mechanism supported by the MPM. During this authentication process, the MPM and the identity mapping service perform the native identity mapping to determine the local identity of the client's network identity. This is important for later authorization since, in a cluster of nodes, there is no concept of a common user space. In other words, on the different nodes in the cluster, some user names may represent the same user, while other user names may represent different users on different hosts.

The identity mapping service uses information stored in the identity mapping files **ctsec_map.global** and **ctsec_map.local**. These identity mapping files are text files containing entries that associate operating system user identifiers on the local system with network security identifiers for authorization purposes. Each node of the cluster has a **ctsec_map.global** file (which contains the common, cluster-wide, identity mappings), and may optionally have a **ctsec_map.local** file which contains identity mappings specific to the local node only.

When the RSCT cluster security services are installed on a node, a default **ctsec_map.global** file is installed. This file contains the default, cluster-wide, identity mapping associations required by RSCT components in order for these systems to execute properly immediately after software installation. There is no default **ctsec_map.local** file.

To modify the cluster-wide identity mappings, or a local node's identity mappings, refer to the instructions in "Configuring the global and local authorization identity mappings" on page 161.

Cluster security services administration

This section describes administrative tasks related to cluster security services. First it discusses the general task of configuring the cluster security services library. Next, in "Configuring the Host Based Authentication mechanism" on page 152, it describes tasks that are specific to Host Based Authentication. Finally, in "Configuring the global and local authorization identity mappings" on page 161, it describes how to modify local and cluster-wide identity mapping configuration files for authorization.

Configuring the cluster security services library

While this section contains information about MPM configuration files, be aware that, since currently only one security mechanism (Host Based Authentication) exists, you should not need to modify this file unless you have moved the Host Based Authentication MPM file to a new location and need to update that location in the configuration file. If you wish to disable Host Based Authentication (even though this effectively eliminates *any* security), contact the IBM Support Center.

Cluster security services provides a Mechanism Abstraction Layer (MAL) that converts the mechanism-independent instructions requested by the application into general tasks to be performed by any mechanism. A Mechanism Pluggable Module (MPM) is a component that converts generalized security services routines into the

specific security mechanism functions. Currently, Host Based Authentication is the only security mechanism supported, and so there is only one MPM (**/usr/lib/unix.mpm**) available at this time.

When cluster security services is installed on a node, a default MPM configuration file is installed in **/usr/sbin/rsct/cfg/ctsec.cfg**. This is an ASCII text file that lists information for each MPM on the system. Since there is only one MPM, there is currently only one entry in the MPM configuration file.

```
#Prior Mnemonic      Code      Path      Flags
#-----
1      unix      0x00001  /usr/lib/unix.mpm  i
```

The entry above contains the path name of the MPM, an identification code number for the MPM, and a priority value. The priority value indicates the preferred security mechanism for the node, and will specify a priority order amongst multiple MPMs when the cluster security services library supports multiple security mechanisms. Since there is only one security mechanism currently supported, the only reason you might need to modify this file is if you have moved the **unix.mpm** file for its default location and wish to indicate the new path. To modify the configuration:

1. Copy the **/usr/sbin/rsct/cfg/ctsec.cfg** file to **/var/ct/cfg/ctsec.cfg**.

```
$ cp /usr/sbin/rsct/cfg/ctsec.cfg /var/ct/cfg/ctsec.cfg
```

Do not modify the default configuration file in **/usr/sbin/rsct/cfg/**.

2. Using an ASCII text editor, modify the new **ctsec.cfg** file in **/var/ct/cfg/**. Do not modify the code, mnemonic, or flag values for this entry.

Configuring the Host Based Authentication mechanism

This section describes the administrative tasks you may need or want to perform that are related to the Host Based Authentication mechanism. The following table outlines the administrative tasks covered.

This task	Describes how to:	Perform this task if:
“Configuring the ctcsd daemon on a node” on page 153	Modify a configuration file read by the Cluster Security Services daemon (ctcsd) upon startup.	You want to modify the operational parameters of the ctcsd daemon. You can configure such things as how many threads the daemon creates, the key generation methods it uses in preparing host public and private keys, and where the daemon looks for key files and the trusted host list.
“Guarding against address and identity spoofing when transferring public keys” on page 157	Copy public keys between nodes to establish the security environment needed for a management domain or an RSCT peer domain.	You do not think your network security is sufficient to prevent address and identity spoofing. If you are confident in the security of your network, you do not need to perform this task; the keys will be copied automatically as part of your node configuration process.
“Changing a node’s private/public key pair” on page 161	Modify a node’s private and public keys.	A node’s private key needs to be modified.

Configuring the ctcsd daemon on a node

When using Host Based Authentication as a security method, cluster security services uses the **ctcsd** daemon to provide and authenticate operating system identity based credentials.

The **ctcsd** daemon obtains its operational parameters from a configuration file (**ctcsd.cfg**). This configuration file instructs the daemon on such things as how many threads to create, the key generation method to use in preparing host public and private keys, where the key files and trusted host lists reside on the node, and whether execution tracing should be enabled.

When cluster security services are installed on a node, a default configuration file is installed in **/usr/sbin/rsct/cfg/ctcsd.cfg**. This is an ASCII text file that contains configurable parameters and their associated default values. **This default configuration file should not be modified.** If you wish to change the **ctcsd** configuration on a node to, for example, improve the performance of the daemon by altering the thread limits, you should:

1. Copy the **/usr/sbin/rsct/cfg/ctcsd.cfg** file to **/var/ct/cfg/ctcsd.cfg**.

```
cp /usr/sbin/rsct/cfg/ctcsd.cfg /var/ct/cfg/ctcsd.cfg
```
2. Using an ASCII text editor, modify the new **ctcsd.cfg** file in **/var/ct/cfg**. The contents of the file will look similar to the following:

```
TRACE= ON
TRACEFILE= /var/ct/IW/log/ctsec/ctcsd/trace
TRACELEVELS= _SEC:Info=1,_SEC:Errors=1
TRACESIZE= 1003520
RQUEUESIZE=
MAXTHREADS=
MINTHREADS=
THREADSTACK= 131072
HBA_USING_SSH_KEYS= false
HBA_PRIVKEYFILE=
HBA_PUBKEYFILE=
HBA_THLFILE=
HBA_KEYGEN_METHOD= rsa512
HBA_CRED_TIMETOLIVE=
SERVICES=hba CAS
```

The keywords listed in this file will set the configurable parameters for the **ctcsd** daemon on this node. The following table describes the configurable parameters.

Table 33. ctcsd daemon configuration file keywords

Keyword	Description
TRACE	<p>Indicates whether or not tracing of the ctcsd daemon is enabled. Valid values are "on" and "off". When tracing is enabled, the TRACEFILE, TRACELEVELS, and TRACESIZE keywords specify the location, level, and size of the trace file generated.</p> <p>Setting the CT_TR_TRACE environment variable overrides any setting specified using the TRACE keyword in the ctcsd.cfg file. For more information about tracing the ctcsd daemon, refer to the <i>Reliable Scalable Cluster Technology for AIX 5L: Diagnosis Guide</i>.</p>

Table 33. *ctcsd* daemon configuration file keywords (continued)

Keyword	Description
TRACEFILE	<p>When tracing of the ctcsd daemon is enabled, this indicates the location of the trace file. If this value is not set, the default location is /var/ct/IW/log/ctsec/ctcsd/trace. The default directory /var/ct/IW/log/ctsec/ctcsd will be created automatically by the ctcsd daemon. However, if you use the TRACEFILE keyword to specify another location, you must ensure that the directory you specify exists. If it does not, the default location will be used instead, and an error will be logged in the trace.</p> <p>Setting the CT_TR_FILENAME environment variable overrides any setting specified using the TRACEFILE keyword in the ctcsd.cfg file. For more information about tracing the ctcsd daemon, refer to the <i>Reliable Scalable Cluster Technology for AIX 5L: Diagnosis Guide</i>.</p>
TRACELEVELS	<p>When tracing of the ctcsd daemon is enabled, the level of the trace. Valid values are:</p> <ul style="list-style-type: none"> _SEC:Info=0 no tracing _SEC:Info=1 trace minimum information messages _SEC:Info=4 trace additional information messages _SEC:Info=8 trace all information messages _SEC:Errors=0 no tracing for errors _SEC:Errors=1 trace all errors causing domain termination _SEC:Errors=2 trace all call errors _SEC:Errors=4 trace failed requests _SEC:Errors=8 trace all errors <p>The trace settings can be combined by using a comma to separate each setting. For example: TRACELEVELS= _SEC:Info=8,_SEC:Errors=8</p> <p>If not specified, the default is _SEC:Info=1, _SEC:Errors=1. Setting the CT_TR_TRACE_LEVELS environment variable overrides any setting specified using the TRACELEVELS keyword in this file. For more information about tracing the ctcsd daemon, refer to the <i>Reliable Scalable Cluster Technology for AIX 5L: Diagnosis Guide</i>.</p>
TRACESIZE	<p>When tracing of the ctcsd daemon is enabled, this indicates the size of the trace file. The minimum size is 4096, and the number specified will be rounded up to the nearest 4096 multiple. If not specified, the default trace-file size is 1003520.</p> <p>Setting the CT_TR_SIZE environment variable overrides any setting specified using the TRACESIZE keyword in the ctcsd.cfg file. For more information about tracing the ctcsd daemon, refer to the <i>Reliable Scalable Cluster Technology for AIX 5L: Diagnosis Guide</i>.</p>

Table 33. *ctcasd* daemon configuration file keywords (continued)

Keyword	Description
RQUEUESIZE	Indicates the maximum length permitted for the daemon's internal run queue. If this value is not set, a default value of 64 is used.
MAXTHREADS	The limit to the number of working threads that the daemon may create and use at any given time (the "high water mark"). If this value is not set, a default value of 10 is used.
THREADSTACK	Sets the internal memory used by the daemon for thread stack space. The value is expressed in bytes. If no value is specified, the default system thread stack size is used. You should not modify this value unless instructed to do so by the IBM Support Center.
MINTHREADS	The number of idle threads that the daemon will retain if the daemon is awaiting further work (the "low water mark"). If this value is not, set, a default value of 4 is used.
HBA_USING_SSH_KEYS	Indicates if the daemon is making use of Secured Remote Shell keys. Acceptable values are true and false. If no value is provided, a default value of false is used. Secured Remote Shell keys are not supported in the current release.
HBA_PRIVKEYFILE	Provides the full path name of the file that contains the local node's private key. The directories in the path must exist. If they do not exist, the ctcasd daemon will terminate. If this value is not set, the default location of <code>/var/ct/cfg/ct_has.qkf</code> is used.
HBA_PUBKEYFILE	Provides the full path name of the file that contains the local node's public key. The directories in the path must exist. If they do not exist, the ctcasd daemon will terminate. If this value is not set, the default location of <code>/var/ct/cfg/ct_has.pkf</code> is used.
HBA_THLFILE	Provides the full path name of the file that contains the local node's trusted host list. If any directory in the path does not exist, the ctcasd daemon will start without creating a trusted host list. If this value is not set, the default location of <code>/var/ct/cfg/ct_has.thl</code> is used.
HBA_KEYGEN_METHOD	Indicates the method to be used by ctcasd to generate the private and public keys of the local node if the files containing these keys do not exist. Acceptable values are those that can be provided as arguments to the ctskeygen -m command. If no value is provided for this attribute, the default value of rsa1024 is used.
HBA_CRED_TIMETOLIVE	Sets the credential life span. The credential life span dictates the period of time after a credential is created that the Host Based Authentication mechanism should consider the credential valid. Setting a credential life span enables the Host Based Authentication mechanism to detect outdated credentials, and refuse authentication to applications presenting such credentials. If no value is specified for this keyword (the default), then credentials will not be checked for expiration. For more information on using this keyword, refer to "Configuring credential life span" on page 156.
SERVICES	Lists the internal library services that the daemon supports. This entry should not be modified by system administrators unless they are explicitly instructed to do so by the IBM Support Center.

3. Stop and restart the **ctcasd** daemon. Be aware that, while the daemon is offline, authentication will not be possible. To stop the daemon, issue the command:

```
stopsrc -s ctcas
```

To restart the daemon, issue the command:

```
startsrc -s ctcas
```

Configuring credential life span

As described in table Table 33 on page 153, the `ctcasd.cfg` file's `HBA_CRED_TIMETOLIVE` keyword sets the credential life span. The credential life span dictates the period of time after a credential is created that the Host Based Authentication mechanism should consider the credential valid. Setting a credential life span enables the Host Based Authentication mechanism to detect outdated credentials, and refuse authentication to applications presenting such credentials. If no value is specified for this keyword (the default), then credentials will not be checked for expiration.

You should not set a credential life span on any node unless all nodes in your cluster have a common agreement on the current time of day. The time of day clocks on all systems within the operational cluster must be set to approximately the same time value. This requirement includes any Hardware Management Consoles (HMCs) that are contained within the operational cluster. Time zone differences between systems are permitted within the cluster, because the time of day clocks measure time in Universal Time Coordinated (UTC).

Sub-second time of day clock synchronization is not necessary for exploiting the credential life span capability of Host Based Authentication. The time of day clocks values need only be set within a reasonable tolerance of each other, typically within a few seconds.

If your cluster makes use of a network time synchronization protocol such as NTP, the nodes of your cluster will already have a common agreement on the current time of day. If you are not using such a protocol, you should use the `date` command on the nodes of your cluster if their time of day clocks do not agree with each other.

The credential life span you specify using the `HBA_CRED_TIMETOLIVE` keyword must allow for time of day clock differences between the systems in the operational cluster, and the workload of these systems. If these factors are not considered when determining the credential life span, it is possible that credentials generated for applications on specific nodes will never be considered valid by specific service applications operating elsewhere within the same cluster. To calculate an appropriate credential life span:

1. Start with the desired credential life span value.
2. Add to this value the largest time of day clock difference between nodes of the operational cluster, including any Hardware Management Consoles (HMCs) in the cluster.
3. Add to this result the largest network latency time known for the nodes within the operational cluster.

For example, say you decide on a credential life span of 30 seconds. If the greatest time of day clock difference (in terms of Universal Time Coordinated) between two nodes is 23 seconds, and the greatest network latency time between any set of systems on the cluster is estimated to be 8 seconds, then the credential life span should be set to 61 seconds.

Once you have decided on the appropriate credential life span, set the `HBA_CRED_TIMETOLIVE` keyword on all systems within the operational cluster with the exception of any Hardware Management Consoles (HMCs) that exist within the cluster. While it is necessary that the time of day clocks on HMCs are set to approximately the same time value as all systems within the operational cluster, it is not necessary to set the `HBA_CRED_TIMETOLIVE` keyword on HMCs.

The default unit of measurement for the time interval specified using the HBA_CRED_TIMETOLIVE keyword is seconds. The time value may be modified using the indicator "m" for minutes and "s" for seconds. The following table show valid specifications for the HBA_CRED_TIMETOLIVE keyword.

This specification:	Specifies a credential life span of:
HBA_CRED_TIMETOLIVE=	(default - infinite)
HBA_CRED_TIMETOLIVE=0	(infinite)
HBA_CRED_TIMETOLIVE=90	(90 seconds)
HBA_CRED_TIMETOLIVE=90s	(90 seconds)
HBA_CRED_TIMETOLIVE=10m	(10 minutes)
HBA_CRED_TIMETOLIVE=10m 15	(10 minutes and 15 seconds)
HBA_CRED_TIMETOLIVE=10m 15s	(10 minutes and 15 seconds)

Guarding against address and identify spoofing when transferring public keys

When configuring a cluster of nodes (either as a management domain using CSM commands or as an RSCT peer domain using configuration resource manager commands), the necessary key exchanges between cluster nodes will, by default, be carried out automatically by CSM or the configuration resource manager.

- In a management domain configured for CSM, the **updatenode** and **installnode** commands will, by default, copy the public key from each of the managed nodes to the management server, and will copy the management server's public key to each of the managed nodes. For more information on the **updatenode** and **installnode** commands, refer to the *IBM Cluster Systems Management: Administration Guide*.
- In an RSCT peer domain, the **preprnode** command, when run on a particular node, will, by default, copy the public key from each of the remote nodes to the local node. Since the command will be run on each node in the domain, each node will have the public key information for all the other nodes in the domain. For information on the **preprnode** command, refer to "Step 1: prepare initial security environment on each node that will participate in the peer domain" on page 26.

Although the commands described above will automatically copy public keys to establish the necessary trust between nodes in the cluster, you must, before using the commands, consider whether the security of the network is sufficient to prevent address and identity spoofing. In a successful spoofing attack on a management domain, for example, a node may allow itself to be managed by the wrong "management server", or the wrong "managed node" may be invited into the network.

If you do not feel your network is sufficiently secure to avoid a possible spoofing attack, you should:

If you are to configure nodes into:	Then you need to:
an RSCT peer domain	manually transfer each node's public key to all other nodes in the RSCT peer domain, and disable the preprnode command's automatic key transferal. Refer to "Manually transferring public keys" on page 158 for more information.

If you are to configure nodes into:	Then you need to:
a management domain	verify the accuracy of the keys automatically transferred by CSM's updatenode and installnode commands. See "Verifying the accuracy of keys that have been automatically transferred" on page 160 for more information.

Manually transferring public keys: In an RSCT peer domain, you will need to copy each node's public key to all other nodes in the domain. To manually transfer public keys:

1. Log on to the node being added to the RSCT peer domain.
2. Determine if the cluster has been set up to use fully qualified host names, short host names, or IP addresses for Host Based Authentication. Make a note of the host name for this node in the corresponding format; this information will be required in Step 5 below.
3. Issue the **ctsvhbal** command to obtain the list of available host based authentication mechanism identities for this system. Output from this command would be similar to:

```
ctsvhbal: The Host Based Authentication (HBA) mechanism identities for the local system are:
```

```
Identity: avenger.pok.ibm.com
```

```
Identity: 9.117.10.4
```

```
ctsvhbal: In order for remote authentication to be successful, at least one of the above identities for the local system must appear in the trusted host list on the remote node where a service application resides. Ensure that at least one host name and one network address identity from the above list appears in the trusted host list on any remote systems that act as servers for applications executing on this local system.
```

4. Obtain the public key for this node by executing the following command:

```
/usr/sbin/rsct/bin/ctskeygen -d > /tmp/hostname_pk.sh
```

This command writes a text version of the local node's public key value to the file **/tmp/hostname_pk.sh**. The contents of this file will consist of two lines of output, similar to the following:

```
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
(generation method: rsa1024)
```

5. Edit the **/tmp/hostname_pk.sh** file, converting it to a shell script that issues the **ctsth1** command to insert this public key into a trusted host list file. Use the host name determined in Step 2 as the argument to the **-n** option. Make sure that the field listed after the generation method field is used as the argument to the **-m** option of this command, and that the text version of the public key is used as the argument to the **-p** option. If the remote node will use a trusted host list file other than the default, list that file's name as an argument to the **-f** option; otherwise, omit the **-f** option. After editing the file, the contents of the file should resemble the following:

```

/usr/sbin/rsct/bin/ctsthl -a -m rsa1024 -n avenger.pok.ibm.com -p
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03

```

- Continue editing the **/tmp/hostname_pk.sh** file. Copy the instruction created in Step 5 above to a new line, and replace the host name argument of the **-n** option with a network address discovered in Step 3. Repeat this process for each network address and host name discovered in Step 3.

Continuing with the previous example, the completed **/tmp/hostname_pk.sh** file would contain:

```

/usr/sbin/rsct/bin/ctsthl -a -m rsa1024 -n avenger.pok.ibm.com -p
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
/usr/sbin/rsct/bin/ctsthl -a -m rsa1024 -n 199.100.100.4 -p
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
/usr/sbin/rsct/bin/ctsthl -a -m rsa1024 -n 9.117.198.45 -p
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03

```

- Transfer the **/tmp/hostname_pk.sh** shell script file to the remote node already within the cluster. This can be done via the **ftp** command, or by transferring this file to a diskette, transferring the diskette to the remote node, and reading the file off the diskette on the remote node.
- Log on to the remote node.
- Execute the **/tmp/hostname_pk.sh** shell script file on the node to add the new node's public key to the node's trusted host list:

```
sh /tmp/hostname_pk.sh
```

- Execute the **/usr/sbin/rsct/bin/ctsthl -l** command to verify that the key has been added to the trusted host list. An example host entry from the trusted host list as it appears in the **ctsthl** command output:

```

-----
Host name: avenger.pok.ibm.com
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
-----
Host name: 199.100.100.4
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
-----
Host name: 9.117.198.45

```

```
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
-----
```

Verifying the accuracy of keys that have been automatically transferred:

When establishing a management domain, CSM's **updatenode** and **installnode** commands will automatically copy:

- the public key from each of the managed nodes to the management server.
- the management server's public key to each of the managed nodes.

If you are concerned about potential address and identity spoofing in a management domain, you will need to verify that that correct keys are copied. To do this:

1. Log on to the node whose public key was copied.
2. Execute the following command on that node:

```
/usr/sbin/rsct/bin/ctskeygen -d > /tmp/hostname_pk.sh
```

This command writes a text version of the local node's public key value to the file **/tmp/hostname_pk.sh**. The contents of this file will consist of two lines of output, resembling the following:

```
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8e15a5dda5
2499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407ccbfc98252072ee1c0
381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3c34e23bb976eb55a386619b70c5
dc9507796c9e2e8eb05cd33cebf7b2b27cf630103
(generation method: rsa1024)
```

3. Log on to the remote node where the key was transferred.
4. Execute the **/usr/sbin/rsct/bin/ctsthl -l** command and verify that the correct key has been added to the trusted host list. The **ctsthl** command output should list entries for the host name and IP address(es) of the node. An example host entry from the trusted host list as it appears in the **ctsthl** command output:

```
-----
Host name: avenger.pok.ibm.com
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
-----
Host name: 199.100.100.4
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
ccbfc98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33cebf7b2b27cf6301
03
-----
Host name: 9.117.198.45
Identifier Generation Method: rsa1024
Identifier Value:
120400cc75f8e007a7a39414492329dcb5b390feacd2bbb81a7074c4edb696bcd8
e15a5dda52499eb5b641e52dbceda2dcc8e8163f08070b5e3fc7e355319a84407
```

```

ccbf98252072ee1c0381bdb23fb686d10c324352329ab0f38a78b437b235dd3d3
c34e23bb976eb55a386619b70c5dc9507796c9e2e8eb05cd33ceb7b2b27cf6301
03
-----

```

Changing a node's private/public key pair

In general, a node's private and public key pair are considered synonymous with a node's identity and are not expected to change over time. However, if they do need to be changed, be aware that a node's private/public key pair should not be changed while a node is operational within the cluster. This is because it is difficult to synchronize a change in a node's public key on all the nodes that need the revised key. The unsynchronized keys will lead to failure in the applications that use cluster security services.

If a node's private key becomes compromised, it is impossible to tell for how long a private key may have been public knowledge or have been compromised. Once it is learned that such an incident has occurred, the system administrator must assume that unwarranted access has been granted to critical system information for an unknown amount of time, and the worst must be feared in this case. Such an incident can only be corrected by a disassembly of the cluster, a reinstall of all cluster nodes, and a reformation of the cluster.

Configuring the global and local authorization identity mappings

As described in "Understanding cluster security services' authorization" on page 150, the identity mapping service uses information stored in the identity mapping files **ctsec_map.global** (which contains the common, cluster-wide, identity mappings) and **ctsec_map.local** (which contains identity mappings specific to the local node only). These are ASCII-formatted files that you can modify using a text editor, thus enabling you to configure the global and local identity mappings.

If you want to create:	Then:
global identity mappings	You need to add entries to the /var/ct/cfg/ctsec_map.global file on every node in the cluster. Entries must not be added to the default /usr/sbin/rsct/cfg/ctsec_map.global file. If the file /var/ct/cfg/ctsec_map.global file does not exist on a node, copy the default /usr/sbin/rsct/cfg/ctsec_map.global file to the /var/ct/cfg directory, and then add the new entries to the /var/ct/cfg/ctsec_map.global file. It is important that you do not remove any entries from the copy file /var/ct/cfg/ctsec_map.global that exist in the default file. It is also important that the /var/ct/cfg/ctsec_map.global files on all nodes within the cluster are identical.
local identity mappings	You need to create, and add entries to, the /var/ct/cfg/ctsec_map.local file on the local node. Be aware that RSCT does not provide a default ctsec_map.local file; you must create it yourself.

When creating **/var/ct/cfg/ctsec_map.global** and **/var/ct/cfg/ctsec_map.local** files, make sure the files can be read by any system user, but that they can be modified only by the root user (or other restrictive user identity not granted to normal system users). By default, these files reside in locally-mounted file systems. While it is possible to mount the **/var/ct/cfg** directory on a networked file system, we discourage this. If the **/var/ct/cfg/ctsec_map.local** file were to reside in a

networked file system, any node with access to that networked directory would assume that these definitions were specific to that node alone when in reality they would be shared.

Each line in the **ctsec_map.global** and **ctsec_map.local** files is an entry. Each entry is used to either associate a security network identifier with a local operating system identifier, or else is used to expressly state that no association is allowed for a particular security network identifier. Lines that start with a pound sign (#) are considered comments and are ignored by the identity mapping service. Blank lines are also ignored by the identity mapping service, so you may include them to improve the readability of the files.

Each entry takes the form:

```
mechanism_mnemonic:identity_mapping
```

Where:

mechanism_mnemonic

is the mnemonic used to represent the security mechanism in the MPM configuration file (as described in “Configuring the cluster security services library” on page 151). Currently, only one security mechanism (Host Based Authentication) exists and is represented by the mnemonic `unix`. All entries will begin with **unix:**.

identity mapping

is either an explicit mapping or a mapping rule. An *explicit mapping* maps a specified security network identifier with a specified local user identifier. A *mapping rule* uses pattern matching and MPM reserved words to determine which security network identifier(s) and local user identifier(s) are mapped.

Both the explicit mappings and the mapping rules can be either affirmative or negative. The *affirmative mappings* are the implied type of mapping; they associate network security identifiers with local user identifiers. The *negative mappings* explicitly state that no association is allowed for one or more network security identifiers.

The exact format of the identity mapping depends on the security mechanism. The MPM that supports the security mechanism can support its own mapping entry format, special characters, and reserved words. Currently only one security mechanism exists (Host Based Authentication). For more information on the format of identity mapping entries for Host Based Authentication, refer to “Configuring the Host Based Authentication mechanism mappings” on page 163.

Since the native identity mapping information is spread out across two files (**ctsec_map.global** and **ctsec_map.local**), it is important to understand how the identity mapping service uses both these files. The identity mapping service parses the **ctsec_map.global** and **ctsec_map.local** files as follows:

1. First, if the **/var/ct/cfg/ctsec_map.local** file exists, the identity mapping service checks for associations in this file.
2. Next, if the **/var/ct/cfg/ctsec_map.global** file exists, the identity mapping service checks for associations in this file.
3. If the **/var/ct/cfg/ctsec_map.global** file does not exist, then the identity mapping service checks for associations in the default file **/usr/sbin/rsct/cfg/ctsec_map.global**.

The identity mapping is performed on a first-match basis. In other words, the first mapping entry for a security network identity (regardless of whether it is an explicit mapping or a mapping rule) is the one applied. For this reason, the order of entries in the mapping file is important; you should place the most restrictive entries before the more relaxed ones. In particular, place entries containing explicit mappings before entries containing mapping rules. Also be aware that, if both the **ctsec_map.global** and **ctsec_map.local** files grant different associations to the same security network identifier, the identity mapping service will use the association stated by the entry in the **ctsec_map.local** file.

Since a single security network identifier may have multiple mapping entries in the mapping file(s), it may not be obvious which mapping is being obtained by the identity mapping service. If authorization is not working as expected, you may want to verify the identity mapping. You can do this using the **ctsidmck** command. The **ctsidmck** command verifies the mapping that would be obtained by the identity mapping service for a specified network identifier. To obtain the mapped identity for the Host Based Authentication mechanism's security network identifier **zathras@greatmachine.epsilon3.org**, you would enter the following at the command prompt:

```
ctsidmck -m unix zathras@greatmachine.epsilon3.org
```

For complete information on the **ctsec_map.global** and **ctsec_map.local** files, and on the **ctsidmck** command, refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Configuring the Host Based Authentication mechanism mappings

To indicate that an entry in the **ctsec_map.global** or **ctsec_map.local** file refers to the Host Based Authentication mechanism, you must begin the entry with **unix:**.

For example:

```
unix:jbrady@epsilon3.ibm.com=jbrady
```

The preceding entry is an example of an affirmative explicit mapping — a specified security network identifier is associated with a specified local user identifier. In this case, the entry associates the Host Based Authentication network identifier **jbrady@epsilon3.ibm.com** to the local user identifier **jbrady**.

To create a negative mapping (a mapping that explicitly states that no association is allowed for a particular security network identifier), use the reserved character **!**. For example, the following entry denies any local user identity association for the Host Based Authentication network identifier **jbrady@epsilon3.ibm.com**.

```
unix:!jbrady@epsilon3.ibm.com
```

Usually, the Host Based Authentication mechanism mappings will use host names as in the preceding examples. However, you can also create mappings using IP address character strings (in IPv4 or IPv6 format). For example, an affirmative explicit mapping might be expressed as:

```
unix:jbrady@9.117.10.14=jbrady
```

Similarly, a negative explicit mapping could be expressed as:

```
unix:!jbrady@9.117.10.14
```

However, you should be aware that IP-address authorization is only possible in an RSCT peer domain in which all nodes are using version 2.3.1.0 (or later) of RSCT. RSCT versions prior to 2.3.1.0 do not support IP-address authorization. In addition, IP-address authorization is not supported in a CSM management domain. In some cluster configurations, authorization might be based on IP addresses for some nodes, and host names for others. In these cases, you might want to create multiple mapping entries for the same host — one using the IP address and one using the host name.

```
unix:jbrady@epsilon2.ibm.com=jbrady
unix:jbrady@9.117.10.14=jbrady
```

Using wildcard characters in Host Based Authentication mappings: You can use the * wildcard character to match multiple user names or host names in the security network identifier. If an entry uses the * wildcard character to match all user names in the security network identifier, it can also use the * wildcard character as the local user identifier. If it does, then the identity mapping service will associate each security network identifier to the local user identifier that matches the user name from the security network identifier. This is the only situation when you can use the * wildcard character in the local user identifier specification. You also cannot use the * wildcard character in place of the security mechanism mnemonic; you must explicitly specify the mnemonic.

For example, the following table shows several examples of how an entry can use the * wildcard character when specifying the user name portion of the security network identifier.

Table 34. Using the wildcard character to match multiple user names in the security network identifier

For example, this entry:	Does this:
unix:*@epsilon3.ibm.com=jbrady	Associates any Host Based Authentication mechanism network identifier from the host epsilon3.ibm.com with the local user identifier jbrady .
unix:!*@epsilon3.ibm.com	Explicitly states that no association is allowed for any Host Based Authentication mechanism network identifier from the host epsilon3.ibm.com .
unix:j*@epsilon3.ibm.com=jbrady	Associates any Host Based Authentication mechanism network identifier starting with the letter "j" from the host epsilon3.ibm.com with the local user identifier jbrady .

The information in the preceding table also applies when you identify a host using its IP address. For example, the entry:

```
unix:*@9.117.10.14=jbrady
```

associates any Host Based Authentication identifier from the host **9.117.10.14** with the local user identifier **jbrady**.

You can only use the * wildcard character once within the user name specification. For example the entry:

```
unix:*athra*@epsilon3.ibm.com=zathras
```

is invalid since the entry repeats the * wildcard character between the token separators : and @.

The following table shows several examples of how an entry can use the * wildcard character when specifying the host identification portion of the security network identifier.

Table 35. Using the wildcard character to match multiple host names in the security network identifier

For example, this entry:	Does this:
unix:jbrady@*=jbrady	Associates any Host Based Authentication mechanism network identifier (host name or IP address) that contains the user name jbrady (regardless of the host) to the local user identifier jbrady .
unix:!jbrady@*	Explicitly states that no association is allowed for any Host Based Authentication mechanism network identifier (host name or IP address) that contains the user name jbrady (regardless of the host).
unix:zathras@*.ibm.com=zathras	Associates any Host Based Authentication mechanism network identifier that contains the user name zathras and a host name ending with the ibm.com network domain to the local user identifier zathras .

When the * wildcard character replaces the entire host identification specification (for example, jbrady@*), it represents any host name or IP address.

You can only use the * wildcard character once within the host identification specification. For example the entry:

```
unix:zathras@*.ibm.*=zathras
```

is invalid since the entry repeats the * wildcard character between the token separators @ and =.

The most powerful use of the * wildcard character is to associate each security network identifier with the local user identifier that matches the user name from the security network identifier. The following table shows several examples of this.

Table 36. Using the wildcard character to associate each security identifier with the local user identifier that matches the user name

For example, this entry:	Does this:
unix:*@epsilon3.ibm.com=*	Associates any Host Based Authentication mechanism network identifier from the host epsilon3.ibm.com to the local user identifier that matches the user name from the security network identifier. For example, zanthras@epsilon3.ibm.com will be associated with the local user identifier zanthras , and jbrady@epsilon3.ibm.com will be associated with the local user identifier jbrady .
unix:*@**	Associates any Host Based Authentication mechanism network identifier (host name or IP address) from any host to the local user identifier that matches the user name from the security network identifier. For example, zanthras@epsilon3.ibm.com will be associated with the local user identifier zanthras , and jbrady@zaphod.ibm.com will be associated with the local user identifier jbrady .

Using MPM-defined reserved words in Host Based Authentication mechanism mappings: In addition to the wildcard character, there are two MPM-defined reserved words you can use when configuring the Host Based Authentication mechanism. These are the **<cluster>** and **<any_cluster>** reserved words.

The **<cluster>** reserved word refers to any host in the currently active cluster. So, for example, the entry:

```
unix:tardis@<cluster>=root
```

will associate any security network identifier that contains the user name **tardis** and originates from any host in the currently active cluster with the local **root** user. For

example, if the hosts **anglashok.ibm.com** and **mimbar.ibm.com** are active in the cluster, then the identity mapping service will associate **tardis@anglashok.ibm.com** and **tardis@mimbar.ibm.com** with the local user **root**.

The **<any_cluster>** reserved word refers to any host within any cluster in which the local node is currently defined. So, for example, the entry:

```
unix:tardis@<any_cluster>=root
```

will associate any security network identifier that contains the user name **tardis** and originates from any host in any cluster in which the local node is defined. For example, if the hosts **anglashok.ibm.com** and **mimbar.ibm.com** are defined within any cluster in which the local node is defined, then the identity mapping service will associate **tardis@anglashok.ibm.com** and **tardis@mimbar.ibm.com** with the local user **root**.

Chapter 7. The Topology Services subsystem

In an RSCT peer domain, the configuration resource manager uses the Topology Services subsystem to monitor the liveness of the adapters and networks included in communication groups. The communication groups are created automatically when you bring the cluster (RSCT peer domain) online (as described in “Step 3: bring the peer domain online” on page 29) or when you explicitly create a group using the **mkcomg** command (as described in “Creating a communication group” on page 44).

This chapter introduces you to the Topology Services subsystem. It:

- includes information about the components of the subsystem, its configuration, other components that depend on it, and how it operates.
- discusses the relationship of the Topology Services subsystem to other subsystems.
- describes a procedure you can use to check the status of the subsystem.

Introducing Topology Services

Topology Services is a distributed subsystem of the IBM Reliable Scalable Cluster Technology (RSCT) software. The RSCT software provides a set of services that support high availability on your system. Another service in the RSCT software is the Group Services distributed subsystem described in Chapter 8, “The Group Services subsystem,” on page 185. Both of these distributed subsystems operate within a domain. A domain is a set of machines upon which the RSCT components execute and, exclusively of other machines, provide their services.

Topology Services provides other high availability subsystems with network adapter status, node connectivity information, and a reliable messaging service. The adapter status and node connectivity information is provided to the Group Services subsystem upon request, Group Services then makes it available to its client subsystems. The Reliable Messaging Service, which takes advantage of node connectivity information to reliably deliver a message to a destination node, is available to the other high availability subsystems.

This adapter status and node connectivity information is discovered by an instance of the subsystem on one node, participating in concert with instances of the subsystem on other nodes, to form a ring of cooperating subsystem instances. This ring is known as a heartbeat ring, because each node sends a heartbeat message to one of its neighbors and expects to receive a heartbeat from its other neighbor. Actually each subsystem instance can form multiple rings, one for each network it is monitoring. This system of heartbeat messages enables each member to monitor one of its neighbors and to report to the heartbeat ring leader, called the Group Leader, if it stops responding. The Group Leader, in turn, forms a new heartbeat ring based on such reports and requests for new adapters to join the membership. Every time a new group is formed, it lists which adapters are present and which adapters are absent, making up the adapter status notification that is sent to Group Services.

In addition to the heartbeat messages, connectivity messages are sent around all rings. Connectivity messages for each ring will forward its messages to other rings, so that all nodes can construct a connectivity graph. It is this graph that determines node connectivity and defines a route that Reliable Messaging would use to send a message between any pair of nodes that have connectivity.

For more detail on maintaining the heartbeat ring and determining node connectivity, see “Topology Services components.”

Topology Services components

The Topology Services subsystem consists of the following components:

Topology Services Daemon

The central component of the Topology Services subsystem.

Pluggable Network Interface Module (NIM)

Program invoked by the Topology Services daemon to communicate with each local adapter.

Port numbers

TCP/IP port numbers that the Topology Services subsystem uses for daemon-to-daemon communications. The Topology Services subsystem also uses UNIX domain sockets for server-to-client and server-to-NIM communication.

Control command

A command that is used to add, start, stop, and delete the Topology Services subsystem, which operates under the SRC subsystem.

Startup command

A command that is used to obtain the configuration from the RSCT peer domain data server and start the Topology Services Daemon. This command is invoked by the SRC subsystem.

Tuning command

A command that is used to change the Topology Services tunable parameters at run-time.

Files and directories

Various files and directories that are used by the Topology Services subsystem to maintain run-time data.

The sections that follow contain more details about each of these components.

The Topology Services daemon (hatsd)

The Topology Services daemon is contained in the executable file `/usr/sbin/rsct/bin/hatsd`. This daemon runs on each node in the RSCT peer domain. Note that the operational domain of the Topology Services subsystem is the RSCT peer domain.

When each daemon starts, it first reads its configuration from a file set up by the Startup command (**cthats**). This file is called the machines list file, because it has all the machines (nodes) listed that are part of the configuration and the IP addresses for each adapter for each of the nodes in that configuration. From this file, the daemon knows the IP address and node number of all the potential heartbeat ring members.

The Topology Services subsystem directive is to form as large a heartbeat ring as possible. To form this ring, the daemon on one node must alert those on the other nodes of its presence by sending a *proclaim* message. According to a hierarchy defined by the Topology Services component, daemons can send a proclaim message only to IP addresses that are lower than its own and can accept a proclaim message only from an IP address higher than its own. Also, a daemon only proclaims if it is the leader of a ring. When a daemon first starts up, it builds a

heartbeat ring for every local adapter, containing only that local adapter. This is called a singleton group and this daemon is the Group Leader in each one of these singleton groups.

To manage the changes in these groups, Topology Services defines the following roles for each group:

Group Leader

The daemon on the node with the local adapter that has the highest IP address in the group. The Group Leader proclaims, handles request for joins, handles death notifications, coordinates group membership changes, and sends connectivity information.

Group Leader Successor

The daemon on the node with the local adapter that has the second highest IP address in the group. This daemon can detect the death of the Group Leader and has the authority to become the Group Leader of the group if that happens.

Mayor A daemon on a node with a local adapter present in this group that has been picked by the Group Leader to broadcast a message to all the adapters in the group. When a daemon receives a message to broadcast, it is a mayor.

Generic

This is the daemon on any node with a local adapter in the heartbeat ring. The role of the Generic daemon is to monitor the heartbeat of the upstream neighbor and inform the Group Leader if the maximum allowed number of heartbeats have been missed.

Each one of these roles are dynamic, which means that every time a new heartbeat ring is formed, the roles of each member are evaluated and assigned.

In summary, Group Leaders send and receive proclaim messages. If the proclaim is from a Group Leader with a higher IP address, then the Group Leader with the lower address replies with a join request. The higher address Group Leader forms a new group with all members from both groups. All members monitor their upstream neighbor for heartbeats. If a sufficient number of heartbeats are missed, a message is sent to the Group Leader and the unresponsive adapter will be dropped from the group. Whenever there is a membership change, Group Services is notified if it asked to be.

The Group Leader also accumulates node connectivity information, constructs a connectivity graph, and routes connections from its node to every other node in the RSCT peer domain. The group connectivity information is sent to all nodes so that they can update their graphs and also compute routes from their node to any other node. It is this traversal of the graph on each node that determines which node membership notification is provided to each node. Nodes to which there is no route are considered unreachable and are marked as down. Whenever the graph changes, routes are recalculated, and a list of nodes that have connectivity is generated and made available to Group Services.

When a network adapter fails or has a problem in one node, this will initially cause incoming heartbeats to be lost. To be able to distinguish a local adapter failure from remote adapter failures, Topology Services will invoke a function which uses *self-death* logic. This self-death logic will attempt to determine whether the adapter is still working. This invokes network diagnosis to determine if the adapter is able to receive data packets from the network. The daemon will try to have data packets

sent to the adapter. If it cannot receive any network traffic, the adapter is considered to be down. Group Services is then notified that all adapters in the group are down.

After an adapter that was down recovers, the daemon will eventually find that the adapter is working again, by using a mechanism similar to the self-death logic, and will form a singleton group with it. This should allow the adapter to form a larger group with the other adapters in the network. An *adapter up* notification for the local adapter is sent to the Group Services subsystem.

Pluggable NIMs

Topology Services pluggable NIMs are processes started by the Topology Services daemon to monitor each local adapter. The NIM is responsible for:

1. Sending messages to a peer daemon upon request from the local daemon.
2. Receiving messages from a peer daemon and forwarding it to the local daemon.
3. Periodically sending heartbeat messages to a destination adapter.
4. Monitoring heartbeats coming from a specified source and notifying the local daemon if any heartbeats are missing.
5. Informing the local daemon if the local adapter goes up or down.

Port numbers and sockets

The Topology Services subsystem uses several types of communications:

- UDP port numbers for intracluster communications, that is, communications between Topology Services daemons within the RSCT peer domain
- UNIX domain sockets for communication between:
 1. The Topology Services clients and the local Topology Services daemon.
 2. The local Topology Services daemon and the NIMs

Intracluster port numbers

For communication between Topology Services daemons within the RSCT peer domain, the Topology Services subsystem uses a single UDP port number. This port number is provided by the configuration resource manager during cluster creation. You supply the UDP port number using the **-t** flag on the **mkrpdomain** command (as described in “Step 2: create a new peer domain” on page 27).

The Topology Services port number is stored in the cluster data so that, when the Topology Services subsystem is configured on each node, the port number is retrieved from the cluster data. This ensures that the same port number is used by all Topology Services daemons in the RSCT peer domain.

This intracluster port number is also set in the **/etc/services** file, using the service name **cthats**. The **/etc/services** file is updated on all nodes in the RSCT peer domain.

UNIX domain sockets

The UNIX domain sockets used for communication are connection-oriented sockets. For the communication between between the Topology Services clients and the local Topology Services daemon, the socket name is **/var/ct/cluster_name/soc/cthats/server_socket**, where *cluster_name* is the name of the RSCT peer domain. For the communication between the local Topology Services daemon and the NIMs, the socket name is

/var/ct/cluster_name/soc/cthats/NIM_name.process_id, where *cluster_name* is the name of the cluster (RSCT peer domain), *NIM_name* is the name of the NIM, and *process_id* is the PID.

The cthatsctrl control command

The Topology Services control command is contained in the executable file **/usr/sbin/rsct/bin/cthatsctrl**. In the normal operation of a cluster, this command should never need to be invoked manually. In fact, in an RSCT peer domain, the configuration resource manager controls the Topology Services subsystem, and using this command directly could yield undesirable results. In an RSCT peer domain, you should use this command only if instructed to do so by IBM service.

The purpose of the **cthatsctrl** command is to add the Topology Services subsystem to the operating software configuration of the cluster. You can also use the command to remove the subsystem from the cluster, start the subsystem, stop the subsystem, and build the configuration file for the subsystem.

The cthats startup command

The Topology Services startup command **cthats** is contained in the executable file **/usr/sbin/rsct/bin/cthats**. The **cthats** command obtains the necessary configuration information from the cluster data server and prepares the environment for the Topology Services daemon. Under normal operating conditions, the Topology Services startup command runs without user initiation. Topology Services is started automatically by the configuration resource manager when you issue the **startpdomain** or **mkcomg** commands. See “Step 3: bring the peer domain online” on page 29 and “Creating a communication group” on page 44 for more information on the **startpdomain** and **mkcomg** commands. However if a problem occurs, the users may need to run the **cthatsctrl** command to operate the Topology Services subsystem.

Note: If using RSCT in conjunction with PSSP running a DCE security environment, the Topology Services startup script will run a conversion program that will convert the DCE key into a cluster compatible key. This will allow nodes running Topology Services using cluster security services to coexist with nodes running Topology Services using DCE.

The cthatstune tuning command

The Topology Services tuning command **cthatstune** is contained in the executable file **/usr/sbin/rsct/bin/cthatstune**. The purpose of the **cthatstune** command is to change the Topology Services’ tunable parameters at runtime. When a communication group is created, Topology Services is, under normally operating conditions, configured with the default values for these parameters or values you supply to the **mkcomg** command. These parameters can be modified using the **chcomg** command as described in “Modifying a communication group’s characteristics” on page 41. You can also use the **cthatstune** command to adjust the parameters directly. The **chcomg** and **cthatstune** commands both allow you to change the parameters without restarting the Topology Services subsystem.

For more information about the **cthatstune** command, refer to its online man page. For complete syntax on all RSCT commands, you can also refer to the *Reliable Scalable Cluster Technology for AIX 5L: Technical Reference* or the *Reliable Scalable Cluster Technology for Linux: Technical Reference*.

Files and directories

The Topology Services subsystem uses the following directories:

- ***/var/ct/cluster_name/log/cthats***, for log files
- ***/var/ct/cluster_name/run/cthats***, for Topology Services daemon current working directory
- ***/var/ct/cluster_name/soc/cthats***, for the UNIX domain socket files.

The ***/var/ct/cluster_name/log/cthats*** (log files)

The ***/var/ct/cluster_name/log/cthats*** directory contains trace output from the Topology Services startup command (***cthats***), Topology Services daemon (***hatsd***), and NIM.

There are four different log files that are created in this directory: the startup command log, the service version of the daemon trace log, the user version of the daemon trace log, and the NIM trace log. The files, each with the same names on all nodes in the cluster, have the following conventions:

1. The Topology Services log from the ***cthats*** startup command is:

cthats.cluster_name[.n]

where:

cluster_name is the name of the cluster to which the node belongs.

n is a number from 1 to 7 with ***cthats.cluster_name.1*** being the most recent instance of the file and ***cthats.cluster_name.7*** being the least recent instance.

The seven most recent instances are kept and older instances are removed.

2. The service version of the log from the ***hatsd*** daemon is:

cthats.DD.HHMMSS.cluster_name

where:

DD is the Day of the Month that this daemon was started.

HHMMSS is the Hour, Minute, and Second that the daemon was started.

cluster_name is the name of the cluster (RSCT peer domain) to which the node belongs.

The contents of this log might be used by IBM Service to help diagnose a problem. The five most recent instances of this file are kept and older instances are removed.

3. The user version of the trace log from the ***hatsd*** daemon is:

cthats.DD.HHMMSS.cluster_name.locale

where:

DD is the Day of the Month that this daemon was started.

HHMMSS is the Hour, Minute, and Second that the daemon was started.

cluster_name is the name of the cluster (RSCT peer domain) to which the node belongs.

locale is the language locale in which the Topology Services daemon was started.

This user version contains error messages that are issued by the ***hatsd*** daemon. The file provides detailed information that can be used together with the syslog for diagnosing problems.

4. The NIM trace log from the pluggable NIM is:

nim.cthats.interface_name.nnn

where:

- *interface_name* is the network interface name. For example, **eth0**.
- *nnn* is a number from 001 to 003

with **nim.cthats.interface_name.001** being the most recent instance of the backup file and **nim.cthats.interface_name.003** the oldest instance. The file without the trailing *nnn* is the current NIM trace log.

The default NIM shipped with Topology Services limits the size of its trace log files to about 200 KB. When the NIM trace log file grows to that limit, the current NIM trace log file is renamed to the most recent back up file and a new NIM trace log file is created. The current and 3 most recent instances of the back up files are kept and the older instances are removed.

The Topology Services daemon limits the size of both the service and user log files to 5,000 lines by default. That limit can be altered by the **cthastune** command. When the limit is reached, the **hatsd** daemon appends the string **.bak** to the name of the current log file and begins a new log file with the same original name. A file that already exists with the **.bak** qualifier is removed before the current log is renamed.

The */var/ct/cluster_name/run/cthats* directory (daemon working files)

The */var/ct/cluster_name/run/cthats* directory is the current working directory for the Topology Services daemon. If the Topology Services daemon abnormally terminates, the core dump file is placed in this directory. Whenever the Topology Services daemon starts, it renames any core file to:

core.DD.HHMMSS.cluster_name

where:

DD is the Day of the Month that the daemon associated with this core file was started.

HHMMSS is the Hour, Minute, and Second that the daemon associated with this core file was started.

cluster_name is the name of the RSCT peer domain to which the node belongs.

The machines list file is also kept in this directory.

The */var/ct/cluster_name/soc/cthats* directory (socket files)

The */var/ct/cluster_name/soc/cthats* directory contains the UNIX domain sockets used for communications between the Topology Services daemon, its clients, and NIMs. The UNIX domain socket name for communications between the Topology Services daemon and its clients is **server_socket**. The UNIX domain socket name for communications between the Topology Services daemon and NIMs is *NIM_name.pid* where:

NIM_name

is the executable name of the NIM. The name of the default NIM shipped with the Topology Services is **default_ip_nim**.

pid is the PID of the NIM process.

Components on which Topology Services depends

The Topology Services subsystem depends on the following components:

System Resource Controller (SRC)

A subsystem feature that can be used to define and control subsystems. The Topology Services subsystem is called **cthats**. The subsystem name is used with the SRC commands (for example, **startsrc** and **lssrc**).

Cluster data

For system configuration information established by the configuration resource manager.

UDP/IP and UNIX-domain socket communication

Topology Services daemons communicate with each other using the UDP/IP sockets. Topology Service daemons communicate with client applications and NIMs using UNIX-domain sockets.

Network adapters

Topology Services will form heartbeat rings on the network.

Cluster security services libraries

The Topology Services subsystem uses the Cluster security services libraries (*libct_mss.a* and *libct_sec.a*) to perform message signature and verification.

First Failure Data Capture (FFDC)

When the Topology Services subsystem encounters events that require system administrator attention, it uses the FFDC facility of RSCT to generate entries in an AIX error log on AIX nodes and the System Log on Linux nodes.

Configuring and operating Topology Services

The following sections describe how the components of the Topology Services subsystem work together to provide topology services. Included are discussions of the following Topology Services tasks:

- Setting Topology Services Tunables
- Configuring Topology Services
- Initializing Topology Services Daemon
- Operating Topology Services

Attention: Under normal operating conditions, Topology Services is controlled by the configuration resource manager. It should not, under normal operating conditions, be necessary to use these Topology Services commands directly. User intervention of Topology Services may cause the configuration resource manager to go down. Exercise caution when operating Topology Services manually.

Setting Topology Services tunables

The cluster data server stores node and network information, as well as some tunable data. The following is a list of the attributes and a brief description of each. Many of these tunables can be set using the **mkcomg** or **chcomg** commands (as described in “Creating a communication group” on page 44 and “Modifying a communication group’s characteristics” on page 41). You can also use the **cthatstune** command (as described in “The cthatstune tuning command” on page 171) to modify Topology Services tunables.

Frequency

Controls how often Topology Services sends a heartbeat to its neighbors. The value is interpreted as the number of seconds between heartbeats. The

minimum and default value is 1. On a system with a high amount of paging activity, this number should be kept as small as possible.

Sensitivity

Controls the number of missed heartbeat messages that will cause a Death in Family message to be sent to the Group Leader. Heartbeats are not considered missing until it has been twice the interval indicated by the Frequency attribute. The default sensitivity value is 4.

Pinning

This controls the memory Pinning strategy. **TEXT** causes the daemon to attempt to pin Text pages, **DATA** attempts to pin Data Pages, **PROC** attempts to pin all pages, and **NONE** causes no pages to be pinned. The default is **PROC**.

The following tunables are available only on AIX nodes.

Run_FixedPri

Run the daemon with a fixed priority. Since Topology Services is a real time application, there is a need to avoid scheduling conflicts. A value of 1 indicates that the daemon is running with fixed priority, 0 indicates that it is not.

FixedPriValue

This is the actual fixed priority level that is used. The daemon will accept values greater than or equal to 10. The default is 38.

Log_Length

This is the approximate number of lines that a log file can hold before it wraps. The default is 5000 lines.

The following tunables are available only on Linux nodes.

Fixed Priority

This is the actual fixed priority level to be used. A value of 0 indicates that the daemon is running at the normal priority level. Linux systems allow fixed priority levels from 1 to 99. The higher the priority level, the higher the precedence for the process to run. Topology Services limits the priority level to a range of between 1 and 80. The default level is 30.

Maximum daemon log file length

This is the approximate number of lines that a log file can hold before it wraps. The default is 5000 lines.

On systems with heavy or unusual load characteristics, it might be necessary to adjust the Frequency and Sensitivity settings. See "Operating Topology Services daemon" on page 177 for more information.

Configuring Topology Services

You may change the default Topology Services configuration options using the **cthatsctrl** command. The **cthatsctrl** command provides a number of functions for controlling the operation of the Topology Services system. You can use it to:

- Add or configure the Topology Services subsystem
- Start the subsystem
- Stop the subsystem
- Delete or unconfigure the subsystem
- "Clean" all Topology Services subsystems

- Turn tracing of the Topology Services daemon on or off
- Refresh (read and dynamically reflect a updated configuration) the subsystem.

Adding the subsystem

The **cthatsctrl** command fetches the port number from the cluster data and places it in the **/etc/services** file. Port numbers are assigned by the configuration resource manager and can be specified when issuing the **mkrpdomain** command. See “Step 2: create a new peer domain” on page 27 for more information on the **mkrpdomain** command.

The second step is to add the Topology Services daemon to the SRC using the **mkssys** command.

On AIX nodes, a third step is to add an entry in the **/etc/inittab** file so that the Topology Services daemon will be started during boot.

Note that if the **cthatsctrl** add function terminates with an error, you can rerun the command after fixing the problem. The command takes into account any steps that already completed successfully.

Starting and stopping the subsystem

The start and stop functions of the **cthatsctrl** command run the **startsrc** and **stopsrc** commands, respectively. However, **cthatsctrl** automatically specifies the subsystem argument to these SRC commands.

Deleting the subsystem

The delete function of the **cthatsctrl** command removes the subsystem from the SRC, and removes the Topology Services daemon communications port number from **/etc/services**. On AIX nodes, the delete function also removes the entry from **/etc/inittab**. The delete function does not remove anything from the cluster data, because the Topology Services subsystem might still be configured on other nodes in the cluster.

Tracing the subsystem

The tracing function of the **cthatsctrl** command is provided to supply additional problem determination information when it is requested by the IBM Support Center. Normally, you should not turn tracing on because it might slightly degrade Topology Services subsystem performance and can consume large amounts of disk space in the **/var** file system.

Initializing Topology Services daemon

Normally, the Topology Services daemon is started by the configuration resource manager when it brings a cluster online. If necessary, you can start the Topology Services daemon using the **cthatsctrl** command or the **startsrc** command directly. The first part of initialization is done by the startup command, **cthats**. It starts the **hatsd** daemon, which completes the initialization steps.

Understanding the initialization process

During this initialization, the startup command does the following:

1. Determines the number of the local node.
2. Obtains the name of the cluster.
3. Retrieves the **machines.lst** file from the local filesystem, where it was placed by the configuration resource manager. The file has identical contents across the active members of the cluster.

4. Performs file maintenance in the log directory and current working directory to remove the oldest log and rename any core files that might have been generated.
5. Starts the Topology Services **hatsd** daemon.

The daemon then continues the initialization with the following steps.

1. Reads the current machines list file and initializes internal data structures.
2. Initializes daemon-to-daemon communication, as well as client communication.
3. Starts the NIMs.
4. For each local adapter defined, forms a membership consisting of only the local adapter.

The daemon is now in its initialized state and ready to communicate with Topology Services daemons on other nodes. The intent is to expand each singleton membership group formed during initialization to contain as many members as possible. Each adapter has an offset associated with it. Only other adapter membership groups with the same offset can join together to form a larger membership group. Eventually, as long as all the adapters in a particular network can communicate with each other, there will be a single group to which all adapters belong.

Merging all adapters into a single group

Initially the subsystem starts out as N singleton groups, one for each node. Each of those daemons is a Group Leader of those singleton groups and knows which other adapters could join the group by the configuration information. The next step is to begin proclaiming to subordinate nodes.

The proclaim logic tries to find members as efficiently as possible. For the first 3 proclaim cycles, daemons proclaim to only their own subnet, and if the subnet is broadcast-capable, that message is broadcast. The result of this is that given the previous assumption that all daemons started out as singletons, this would evolve into M groups, where M is the number of subnets that span this heartbeat ring. On the fourth proclaim cycle, those M Group Leaders send proclaims to adapters that are outside of their local subnet. This will cause a merging of groups into larger and larger groups until they have coalesced into a single group.

From the time the groups were formed as singletons until they reach a stabilization point, the groups are considered unstable. The stabilization point is reached when a heartbeat ring has no group changes for the interval of 10 times the heartbeat send interval. Up to that point, the proclaim continues on a 4 cycle operation, where 3 cycles only proclaim to the local subnets, and one cycle proclaims to adapters not contained on the local subnet. After the heartbeat ring has reached stability, proclaim messages go out to all adapters not currently in the group regardless of the subnet to which they belong. Adapter groups that are unstable are not used when computing the node connectivity graph.

Operating Topology Services daemon

Normal operation of the Topology Services subsystem does not require administrative intervention. The subsystem is designed to recover from temporary failures, such as node failures or failures of individual Topology Services daemons. Topology Services also provides indications of higher level system failures. However, there are some operational characteristics of interest to system administrators and after adding or removing nodes or adapters, you might need to refresh the subsystem.

Defaults and limitations

The maximum node number allowed is 2047. The maximum number of networks it can monitor is 48.

Topology Services is meant to be sensitive to network response and this sensitivity is tunable. However, other conditions can degrade the ability of Topology Services to accurately report on adapter or node membership. One such condition is the failure to schedule the daemon process in a timely manner. This can cause daemons to be late in sending their heartbeats by a significant amount. This can happen because an interrupt rate is too high, the rate of paging activity is too high, or there are other problems. If the daemon is prevented from running for enough time, the node might not be able to send out heartbeat messages and will be considered, incorrectly, to be down by other peer daemons.

Since Topology Services is a real time process, do not intentionally subvert its use of the CPU because you can cause false indications.

On AIX nodes, Topology Services sets all four of the following options to 1 so that the reliable message feature which utilizes IP source routing, will continue to work. Disabling any of these network options can prevent the reliable message feature from working properly.

- **ipsrctestsend** (default is 1)
- **ipsrctestrecv** (default is 0)
- **ipsrctestforward** (default is 1)
- **nonlocsrcroute** (default is 0)

ATTENTION - READ THIS FIRST

The network options to enable IP source routing are set to their default values for security reasons. Since changing them may cause the node to be vulnerable to network attack, system administrators are advised to use other methods to protect the cluster from network attack.

Topology Services requires the IP source routing feature to deliver its data packets when the networks are broken into several network partitions. The network options must be set correctly to enable the IP source routing. On Linux Systems, the Topology Services startup command will set the options:

IP forward: enable

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Accept Source Routing: enable

```
echo 1 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

```
echo 1 > /proc/sys/net/ipv4/conf/interface/accept_source_route
```

RP Filter: disable

```
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```

```
echo 0 > /proc/sys/net/ipv4/conf/interface/rp_filter
```

Dependency on lo0 interface: Correct operation of Topology Services depends on the loopback network interface (lo0) being defined and in the **up** state. If the loopback network interface is disabled, Topology Services will be unable to transmit

Group Services through non-IP devices. Group Services messages are sent through non-IP interfaces when no IP communication paths exist between a given pair of nodes.

Tuning the Topology Services subsystem

The default settings for the frequency and sensitivity tunable attributes discussed in “Configuring Topology Services” on page 175 are overly aggressive for clusters that have more than 128 nodes or heavy load conditions. Using the default settings will result in false failure indications. Decide which settings are suitable for your system by considering the following:

- Higher values for the frequency attribute result in lower CPU and network utilization from the Topology Services daemon. Higher values for the product of frequency times sensitivity result in less sensitivity of Topology Services to factors that cause the daemon to be blocked or messages to not reach their destinations. Higher values for the product also result in Topology Services taking longer to detect a failed adapter or node.
- If the nodes are used primarily for parallel scientific jobs, use the following settings:

Frequency	Sensitivity	Seconds to detect node failure
2	6	24
3	5	30
3	10	60
4	9	72

- If the nodes are used in a mixed environment or for database workloads, use the following settings:

Frequency	Sensitivity	Seconds to detect node failure
2	6	24
3	5	30
2	10	40

- If the nodes tend to operate in a heavy paging or I/O intensive environment, use the following settings:

Frequency	Sensitivity	Seconds to detect node failure
1	12	24
1	15	30

By default Topology Services uses:

Frequency	Sensitivity	Seconds to detect node failure
1	4	8

You can adjust the tunable attributes by using the **chcomg** command (as described in “Modifying a communication group’s characteristics” on page 41). You can also use the **cthatstune** command. For example, to change the frequency attribute to the value 2 on network **en_net_0** and then refresh the Topology Services subsystem, use the command:

```
cthatstune -f en_net_0:2 -r
```

EtherChannel and IEEE 802.3ad Link Aggregation considerations for AIX 5L:

On AIX 5L machines, EtherChannel and IEEE 802.3ad Link Aggregation are network port aggregation technologies that allow several Ethernet adapters to be

aggregated together to form a single pseudo Ethernet device. For example, *ent0* and *ent1* can be aggregated to *ent3*; interface *en3* would then be configured with an IP address. The system considers these aggregated adapters as one adapter. Therefore, IP is configured over them as over any Ethernet adapter. In addition, all adapters in the EtherChannel or Link Aggregation are given the same hardware (MAC) address, so they are treated by remote systems as if they were one adapter.

The main benefit of EtherChannel and IEEE 802.3ad Link Aggregation is that they have the network bandwidth of all of their adapters in a single network presence. In addition, if an adapter fails, the packets are automatically sent on the next available adapter without disruption to existing user connections. The adapter is automatically returned to service on the EtherChannel or Link Aggregation when it recovers.

Details on how to configure EtherChannel and IEEE 802.3ad Link Aggregation are provided in the *AIX 5L System Management Guide: Communications and Networks* online manual. Refer to the following URL for this information:

http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/commadm/tcp_etherchannel.htm

If the preceding URL no longer points to an active Web page, refer to the URL **<http://www.ibm.com/servers/aix/library/>** for AIX documentation.

The link aggregation technologies provide quick detection and recovery from adapter failures. Once a given adapter fails, other adapters which are part of the aggregation will take over IP communication within around 4 seconds.

When the adapter problem is fixed, the adapter gets reactivated into the aggregation. At that point, a disruption of around 8 seconds in IP communication may occur. This disruption is caused by the adapter being declared “fit for use” by AIX while the switch is still evaluating the new topology. The duration of the disruption may depend on the brand and size of the switch, and may be reduced by configuring the switch not to use “Spanning Tree Protocol”.

Because adapter failure and recovery may lead to short-term communication outages, RSCT needs to be tuned to allow for a longer adapter detection time. Without tuning, false failure indications may occur during the outages.

The values to be tuned are the Topology Services “heartbeat frequency” and “heartbeat sensitivity”. The exact value to be used depends on the length of the communication outage, which itself depends on factors such as adapter type, and brand and size of the switch. A good initial set of values is one that results in detection time around 16 seconds.

It is suggested that experiments be performed to determine how lengthy the outages are for a given system configuration. The experiments should consist of pulling adapter cables and then reconnecting them after a few minutes. If error log entries of type `TS_LOC_DOWN_ST` or `TS_DEATH_TR` are generated (assuming that RSCT is running when the experiments are attempted), then this is an indication that the adapter detection tunables need to be increased. To help determine the length of the outages, a sequence such as the following can be run:

```
while:
  do date
    ping -w1 -c1 <IP address>
    sleep 1
  done
```

The interval during which the packets are lost ("100% packet loss" seen in the output) determines for how long communication with the aggregation was not available.

Network Interface Backup (NIB): EtherChannel Backup is a variant of EtherChannel that is used for high-availability only. EtherChannel Backup allows an aggregated adapter to have a backup. If all adapters that compose the aggregation fail, then communication is switched to the backup adapter until any adapter in the main channel recovers. A variant of it is Network Interface Backup (NIB): in this mode of operation, there is only 1 adapter in the main channel and a backup adapter. While NIB by itself does not provide better bandwidth than the physical adapter, it can be used to work around switch failures. Usually port aggregation requires all adapters to be connected to the same switch, which makes the switch the single point of failure. By using NIB, and by connecting the primary and backup adapters to different switches, communication will not be lost by the failure of a single switch.

To help detect loss of network reachability (in addition to detecting failures in the adapter and its connection to the switch), NIB allows specifying an address to be pinged. If the given address cannot be reached after a given number of attempts (both specified when NIB is defined), then the current "active" adapter is considered down, resulting in the backup adapter taking over communication. Setting reasonable values for the "Number of Retries" option is important to ensure smooth operation of NIB: if the value is not enough to cover the period during which the switch is reconfiguring itself, it is likely that there will be multiple (false) takeover operations until one of the adapters becomes the owner of the aggregation. Such extra takeover activity makes real (or desired) takeover operations take much longer than intended.

As an initial guideline, setting "Number of Retries" to 10 should correct the false takeover problem in cases where communication outages are around 8 seconds.

The "false takeover" scenario can be identified by examining the AIX error log. In case the scenario occurs, entries like the following may appear:

- ECH_PING_FAIL_PRMRY
- ECH_PING_FAIL_BCKP
- GXENT_LINK_DOWN

When "Number of Retries" is set to an adequate value, then error log entry ECH_CHAN_FAIL may be the only one to be generated.

Since NIB uses a single adapter as primary, an EtherChannel-enabled switch is not required.

Refreshing the Topology Services daemon

In an RSCT peer domain, all refresh operations should occur without user intervention. The Topology Services subsystem needs to be refreshed before it can recognize a new configuration. However, if you need to manually refresh the Topology Services subsystem, run either the **cthatctrl** command or the **cthatstune** command both with the **-r** option on any node in the cluster.

Note that if there are nodes in the cluster that are unreachable with Topology Services active, they will not be refreshed. Also, if the connectivity problem is resolved such that Topology Services on that node is not restarted, the node

refreshes itself to remove the old configuration. Otherwise, it will not acknowledge nodes or adapters that are part of the configuration, but not in the old copy of the configuration.

Topology Services procedures

Normally, the Topology Services subsystem runs itself without requiring administrator intervention. On occasion, you might need to check the status of the subsystem.

Displaying the status of the Topology Services daemon

You can display the operational status of the Topology Services daemon by issuing the **lssrc** command. Topology Services monitors the networks that correspond to the communication groups set up by the configuration resource manager. To see the status of the networks you need to run the command on a node that is up:

lssrc -ls cthats

In response, the **lssrc** command writes the status information to the standard output. The information includes:

- The information provided by the **lssrc -s cthats** command (short form).
- Seven lines for each network for which this node has an adapter and includes the following information:
 - The network name.
 - The network index.
 - The number of defined members, number of adapters that the configuration reported existing for this network.
 - The number of members, number of adapters currently in the membership group.
 - The state of the membership group, denoted by S (Stable), U (Unstable), or D (Disabled).
 - Adapter ID, the address and instance number for the local adapter in this membership group.
 - Group ID, the address and instance number of the membership group. The address of the membership group is also the address of the group leader.
 - Adapter interface name.
 - HB Interval, which corresponds to the **Frequency** attribute in the cluster. This exists both on a per network basis and a default value which could be different.
 - HB Sensitivity, which corresponds to the **Sensitivity** attribute in the cluster. This exists both on a per network basis and a default value which could be different.
 - The total number of missed heartbeats detected by the local adapter, and the total number in the current instance of the group.
 - Two lines of the network adapter statistics.
 - The PID of the NIMs.
- The number of clients connected and the client process IDs and command names.
- Configuration Instance, the Instance number of the Machines List file.
- Whether the daemon is using message authentication. If it is, the version number of the key used for mutual authentication is also included.

- The size of the data segment of the process and the number of outstanding allocate memory without corresponding free memory operations.
- The segments pinned. **NONE**, a combination of **TEXT**, **DATA**, and **STACK**, or **PROC** .
- The size of text, static data, and dynamic data segments. Also, the number of outstanding memory allocations without a corresponding free memory operation.
- Whether the daemon is processing a refresh request.
- Daemon process CPU time, both in user and kernel modes.
- The number of page faults and the number of times the process has been swapped out.
- The number of nodes that are seen as reachable (up) from the local node and the number of nodes that are seen as not reachable (down).
- A list of nodes that are either up or down, whichever list is smaller. The list of nodes that are down includes only the nodes that are configured and have at least one adapter which Topology Services monitors. Nodes are specified in the list using the format:

N1–N2(I1) N3–N4(I2)...

where *N1* is the initial node in a range, *N2* is the final node in a range, and *I1* is the increment. For example, 5–9(2) specifies nodes 5, 7, and 9. If the increment is 1 then the increment is omitted. If the range has only one node, only the one node number is specified.

The following is an example of the output from the **lssrc -ls cthats** command on a node:

```
Subsystem      Group      PID      Status
cthats         cthats     827      active
Network Name   Indx Defd Mbrs St Adapter ID      Group ID
en_net_0       [ 0]    3    2  S 9.114.67.72     9.114.67.73
en_net_0       [ 0]  eth0          0x32c37ded      0x32c3907b
HB Interval = 1 secs. Sensitivity = 4 missed beats
Missed HBs: Total: 10 Current Group: 2
Packets sent   : 4706 ICMP 0 Errors: 0 No mbuf: 0
Packets received: 3537 ICMP 0 Dropped: 0
NIM's PID: 884
  1 locally connected Client with PID:
hagsd( 907)
  Configuration Instance = 1244520230
  Default: HB Interval = 1 secs. Sensitivity = 4 missed beats
  Daemon employs no security
  Segments pinned: Text Data Stack.
  Text segment size: 548 KB. Static data segment size: 486 KB.
  Dynamic data segment size: 944. Number of outstanding malloc: 88
  User time 3 sec. System time 1 sec.
  Number of page faults: 1245. Process swapped out 0 times.
  Number of nodes up: 2. Number of nodes down: 1.
  Nodes down : 1
```

The network being monitored in the last example is named **en_net_0**. The **en_net_0** network has 3 adapters defined and 2 of them are members of the group. The group is in stable state. The frequency and sensitivity of this network is 1 second and 4 missing heartbeats respectively. Currently, there is only one client, **hagsd**. The total number of missed heartbeats detected by the local adapter is 10, and the total number in the current instance of the group is 2. All text, data, and stack segments are pinned in the main memory. There are 2 nodes up and 1 node down. The down node is node 1.

Chapter 8. The Group Services subsystem

The configuration resource manager uses the Group Services subsystem to provide distributed coordination, messaging, and synchronization among nodes in an RSCT peer domain. When issuing the **startprdomain** command to bring a cluster (RSCT peer domain) online, the configuration resource manager will, if necessary, start Group Services. Under normal operating conditions, it will not be necessary for you to directly influence Group Services.

This chapter introduces you to the Group Services (GS) subsystem. It:

- includes information about the component of the subsystem, its configuration, other components on which it depends, and how it operates.
- discusses the relationship of the Group Services subsystem to the other high availability subsystems.
- describes a procedure you can use to check the status of the subsystem.

Introducing Group Services

Group Services is a distributed subsystem of the IBM Reliable Scalable Cluster Technology (RSCT) software. RSCT software provides a set of services that support high availability on your system. Another service included with the RSCT software is the Topology Services distributed subsystem. The Topology Services subsystem is described in Chapter 7, “The Topology Services subsystem,” on page 167.

The function of the Group Services subsystem is to provide other subsystems with a distributed coordination and synchronization service. These other subsystems that depend upon Group Services are called *client subsystems*. Each client subsystem forms one or more *groups* by having its processes connect to the Group Services subsystem and use the various Group Services interfaces. A process of a client subsystem is called a *GS client*.

A group consists of two pieces of information:

- The list of processes that have joined the group, called the *group membership list*.
- A client-specified *group state value*.

Group Services guarantees that all processes that are joined to a group see the same values for the group information, and that they see all changes to the group information in the same order. In addition, the processes may initiate changes to the group information via *protocols* that are controlled by Group Services.

A GS client that has joined a group is called a *provider*. A GS client that wishes only to monitor a group, without being able to initiate changes in the group, is called a *subscriber*.

Once a GS client has initialized its connection to Group Services, it can join a group and become a provider. All other GS clients that have already joined the group (those that have already become providers) are told as part of a join protocol about the new providers that wish to join. The existing providers can either accept new joiners unconditionally (by establishing a one-phase join protocol) or vote on the protocol (by establishing an n-phase protocol). During a vote, they can choose to

approve the protocol and accept the new providers into the group, or reject the protocol and refuse to allow the new providers to join.

Group Services monitors the status of all the processes that are joined to a group. If either the process or the node on which a process is executing fails, Group Services initiates a failure protocol that informs the remaining providers in the group that one or more providers have been lost.

Join and failure protocols are used to modify the membership list of the group. Any provider in the group may also propose protocols to modify the state value of the group. All protocols are either unconditional (one-phase) protocols, which are automatically approved and not voted on, or conditional (n-phase) protocols, which are voted on by the providers.

During each phase of an n-phase protocol, each provider can take application-specific action and *must* vote to approve, reject, or continue the protocol. The protocol completes when it is either approved (the proposed changes become established in the group), or rejected (the proposed changes are dropped).

Group Services components

The Group Services subsystem consists of the following components:

Group Services daemon

The central component of the Group Services subsystem.

Group Services API (GSAPI)

The application programming interface that GS clients use to obtain the services of the Group Services subsystem.

Port numbers

TCP/IP port numbers that the Group Services subsystem uses for communications. The Group Services subsystem also uses UNIX domain sockets.

Control command

A shell command that is used to add, start, stop, and delete the Group Services subsystem, which operates under control of the SRC. On Linux, SRC is an RSCT subsystem. On AIX, it is a component of the operating system.

Files and directories

Various files and directories that are used by the Group Services subsystem to maintain run-time data.

The sections that follow contain more details about each of these components.

The Group Services daemon (hagsd)

The Group Services daemon is contained in the executable file `/usr/sbin/rsct/bin/hagsd`. This daemon runs on each node in the peer domain

A GS client communicates with a Group Services daemon that is running on the same node as the GS client. A GS client communicates with the Group Services daemon, through the GSAPI software, using a UNIX domain socket. For HACMP™, before a GS client registers with Group Services, it must set the `HA_DOMAIN_NAME` and the `HA_GS_SUBSYS` environment variables to the

HACMP cluster name and "grpsvcs" respectively. In an RSCT peer domain, the **HA_DOMAIN_NAME** and the **HA_GS_SUBSYS** environment variables **should not** be set.

The Group Services API (GSAPI)

The Group Services Application Programming Interface (GSAPI) is a shared library that a GS client uses to obtain the services of the Group Services subsystem. This shared library is supplied in two versions: one for non-thread-safe programs and one for thread-safe programs. These libraries are referenced by the following path names:

	On Linux Nodes:	On AIX Nodes:
Non-Thread-Safe Version	/usr/lib/libha_gs.so	/usr/lib/libha_gs.a
Thread-Safe Version	/usr/lib/libha_gs_r.so	/usr/lib/libha_gs_r.a

The path names shown in the preceding table are symbolic links to the actual files located in **/usr/sbin/rsct/lib**. For serviceability, these libraries are supplied as shared libraries.

For details on the GSAPI software, see the *Group Services Programming Guide and Reference*.

To allow non-root users to use Group Services:

1. Create a group named **hagsuser**.
2. Add the desired user IDs to the **hagsuser** group.
3. Stop and restart **cthags** (if it was running before you created the **hagsuser** group).

Users in the created **hagsuser** group can use Group Services.

Port numbers and sockets

The Group Services subsystem uses several types of communications:

- UDP port numbers for intra-domain communications, that is, communications between Group Services daemons within an operational domain which is defined within the cluster.
- UNIX domain sockets for communication between GS clients and the local Group Services daemon (via the GSAPI).

Intra-domain port numbers

For communication between Group Services daemons within an operational domain, the Group Services subsystem uses a single UDP port number. This port number is provided by the configuration resource manager during cluster creation. You supply the port number using the **-g** flag on the **mkrpdomain** command (as described in "Step 2: create a new peer domain" on page 27).

The Group Services port number is stored in the cluster data so that, when the Group Services subsystem is configured on each node, the port number is fetched from the cluster data. This ensures that the same port number is used by all Group Services daemons in the same operational domain within the cluster.

This intra-domain port number is also set in the **/etc/services** file, using the service name **cthags**. The **/etc/services** file is updated on all nodes in the cluster.

UNIX domain sockets

UNIX domain sockets are used for communication between GS clients and the local Group Services daemon (via the GSAPI). These are connection-oriented sockets. The socket name used by the GSAPI to connect to the Group Services daemon is */var/ct/cluster_name/soc/hagsdsocket*.

The cthagsctrl control command

The Group Services control command is contained in the executable file */usr/sbin/rsct/bin/cthagsctrl*.

The purpose of the **cthagsctrl** command is to add (configure) the Group Services subsystem to the cluster. It can also be used to remove the subsystem from the cluster; and start and stop the subsystem. Normally, you will not need to issue this command directly. In fact, in an RSCT peer domain, the configuration resource manager controls the Group Services subsystem, and using this command directly could yield undesirable results. In an RSCT peer domain, you should use this command only if instructed to do so by IBM service.

For more information, see “Configuring Group Services” on page 189.

Files and directories

The Group Services subsystem uses the following directories:

	On Linux Nodes:	On AIX Nodes:
Lock Files	<i>/var/ct/cluster_name/lck</i>	<i>/var/ct/cluster_name/lck/cthags</i>
Log Files	<i>/var/ct/cluster_name/log</i>	<i>/var/ct/cluster_name/log/cthags</i>
Working Directory for the Group Services Daemon	<i>/var/ct/cluster_name/run</i>	<i>/var/ct/cluster_name/run/cthags</i>
Socket Files	<i>/var/ct/cluster_name/soc</i>	<i>/var/ct/cluster_name/soc/cthags</i>

Lock files

On Linux nodes, lock files are located in */var/ct/cluster_name/lck*. On AIX nodes, lock files are located in */var/ct/cluster_name/lck/cthags*. In the lock file directory, **cthags.tid**. is used to ensure a single running instance of the Group Services daemon, and to establish an instance number for each invocation of the daemon.

Log files

On Linux nodes, log files are located in */var/ct/cluster_name/log*. On AIX nodes, log files are located in */var/ct/cluster_name/log/cthags*. The log file directory contains trace output from the Group Services daemon.

On the nodes, the files are called **cthags_nodenum_instnum.cluster_name**, **cthags_nodenum_instnum.cluster_name.long**, and **cthags.default.nodenum_instnum**, where:

- *nodenum* is the node number on which the daemon is running
- *instnum* is the instance number of the daemon.

The Group Services daemon limits the log size to a pre-established number of lines (by default, 5,000 lines). When the limit is reached, the daemon appends the string **.bak** to the name of the current log file and begins a new log. If a **.bak** version already exists, it is removed before the current log is renamed.

Working directory for Group Services daemon

On Linux, the working directory for the Group Services daemon is */var/ct/cluster_name/run*. On AIX nodes, it is */var/ct/cluster_name/run/cthags*. If the Group Services daemon abnormally terminates, the core dump file is placed in this working directory. Whenever the Group Services daemon starts, it renames any core file to **core_nodenum.instnum**, where *nodenum* is the node number on which the daemon is running and *instnum* is the instance number of the previous instance of the daemon.

Components on which Group Services depends

The Group Services subsystem depends on the following components:

System Resource Controller (SRC)

A subsystem that can be used to define and control subsystems. The Group Services subsystem is called **cthags**. The subsystem name is used with the SRC commands (for example, **startsrc** and **lssrc**).

Cluster data

For system configuration information established by the configuration resource manager.

Topology Services

A subsystem that is used to determine which nodes in a system can be reached (that is, are running) at any given time. It is often referred to as **heartbeat**. The Topology Services subsystem is SRC-controlled. It is called **cthats**. For more information, see Chapter 7, “The Topology Services subsystem,” on page 167.

UDP/IP and UNIX-domain socket communication

Group Services daemons communicate with each other using the UDP/IP feature sockets. Topology Service daemons communicate with client applications using UNIX-domain sockets.

First Failure Data Capture (FFDC)

When the Group Services subsystem encounters events that require system administrator attention, it uses the FFDC facility of RSCT to generate entries in a syslog.

Configuring and operating Group Services

The following sections describe how the components of the Group Services subsystem work together to provide group services. Included are discussions of Group Services:

- Configuration
- Daemon initialization and errors
- Operation

Configuring Group Services

Group Services configuration is performed by the **cthagsctrl** command, which is invoked by the configuration resource manager. Under normal operating conditions, you will not need to directly invoke this command. In fact, doing so could yield undesirable results. In an RSCT peer domain, you should use this command only if instructed to do so by IBM service.

The **cthagsctrl** command provides a number of functions for controlling the operation of the Group Services system. You can use it to:

- Add (configure) the Group Services subsystem
- Start the subsystem
- Stop the subsystem
- Delete (unconfigure) the subsystem
- Clean all Group Services subsystems
- Turn tracing of the Group Services daemon on or off

Adding the subsystem

The **cthagsctrl** command fetches the port number from the cluster data.

The second step is to add the Group Services daemon to the SRC using the **mkssys** command.

Note that if the **cthagsctrl** add function terminates with an error, the command can be rerun after the problem is fixed. The command takes into account any steps that already completed successfully.

Starting and stopping the subsystem

The start and stop functions of the **cthagsctrl** command simply run the **startsrc** and **stopsrc** commands, respectively. However, **cthagsctrl** automatically specifies the subsystem argument to these SRC commands.

Deleting the subsystem

The delete function of the **cthagsctrl** command removes the subsystem from the SRC, and removes the Group Services daemon communications port number from **/etc/services**. It does *not* remove anything from the cluster data, because the Group Services subsystem may still be configured on other nodes in the operational domain.

Cleaning the subsystem (AIX only)

On AIX, the clean function of the **cthagsctrl** command performs the same function as the delete function, except in all system partitions. In addition, it removes the Group Services daemon remote client communications port number from the **/etc/services** file.

The clean function does *not* remove anything from the cluster data. This function is provided to support restoring the system to a known state, where the known state is in the cluster data.

Tracing the subsystem

The tracing function of the **cthagsctrl** command is provided to supply additional problem determination information when it is requested by the IBM Support Center. Normally, tracing should *not* be turned on, because it might slightly degrade Group Services subsystem performance and can consume large amounts of disk space in the **/var** file system.

Initializing Group Services daemon

Normally, the Group Services daemon is started by the configuration resource manager when it brings a cluster (RSCT peer domain) online. If necessary, the Group Services daemon can be started using the **cthagsctrl** command or the **startsrc** command directly.

During initialization, the Group Services daemon performs the following steps:

1. It gets the number of the node on which it is running. On AIX, the Group Services daemon gets this information from the local peer domain configuration. On Linux, the Group Services daemon gets this information from the cluster definition file which was configured during the RSCT configuration.
2. It tries to connect to the Topology Services subsystem. If the connection cannot be established because the Topology Services subsystem is not running, it is scheduled to be retried every 20 seconds. This continues until the connection to Topology Services is established. Until the connection is established, the Group Services daemon writes an error log entry periodically and no clients may connect to the Group Services subsystem.
3. It performs actions that are necessary to become a daemon. This includes establishing communications with the SRC subsystem so that it can return status in response to SRC commands.
4. It establishes the Group Services domain, which is the set of nodes in the cluster.
 At this point, one of the GS daemons establishes itself as the GS nameserver. For details, see “Establishing the GS nameserver.”
 Until the domain is established, no GS client requests to join or subscribe to groups are processed.
5. It enters the main control loop.
 In this loop, the Group Services daemon waits for requests from GS clients, messages from other Group Services daemons, messages from the Topology Services subsystem, and requests from the SRC for status.

Establishing the GS nameserver

The Group Services subsystem must be able to keep track of the groups that its clients want to form. To do this, it establishes a GS nameserver within the domain. The GS nameserver is responsible for keeping track of all client groups that are created in the domain.

To ensure that only one node becomes a GS nameserver, Group Services uses the following protocol:

1. When each daemon is connected to the Topology Services subsystem, it waits for Topology Services to tell it which nodes are currently running.
2. Based on the input from Topology Services, each daemon finds the lowest-numbered running node in the domain. The daemon compares its own node number to the lowest-numbered node and performs one of the following:
 - If the node the daemon is on is the lowest-numbered node, the daemon waits for all other running nodes to nominate it as the GS nameserver.
 - If the node the daemon is on is not the lowest-numbered node, it sends nomination messages to the lowest-numbered node periodically, initially every 5 seconds.
3. Once all running nodes have nominated the GS nameserver-to-be and a coronation timer (about 20 seconds) has expired, the nominee sends an insert message to the nodes. All nodes must acknowledge this message. When they do, the nominee becomes the established GS nameserver, and it sends a commit message to all of the nodes.
4. At this point, the Group Services domain is established, and requests by clients to join or subscribe to groups are processed.

Note that this description is in effect when all nodes are being booted simultaneously, such as at initial system power-on. It is often the case, however, that a Group Services daemon is already running on at least one node and is

already established as the domain's GS nameserver. In that case, the GS nameserver waits only for Topology Services to identify the newly running nodes. The GS nameserver will then send the newly running nodes proclaim messages that direct the nodes to nominate it as nameserver. Once those nodes then nominate the GS nameserver, the GS nameserver simply executes one or more insert protocols to insert the newly-running nodes into the domain.

Group Services initialization errors

The Group Services subsystem creates error log entries to indicate severe internal problems. For most of these, the best response is to contact the IBM Support Center.

However, if you get a message that there has been no heartbeat connection for some time, it could mean that the Topology Services subsystem is not running.

To check the status of the Topology Services subsystem, issue the **lssrc -l -s cthags** command. If the response indicates that the Topology Services subsystem is inoperative, try to restart it using the **startprdomain** or **startprnode** command. If you are unable to restart it, call the IBM Support Center.

Group Services daemon operation

Normal operation of the Group Services subsystem requires no administrative intervention. The subsystem normally recovers from temporary failures, such as node failures or failures of Group Services daemons, automatically. However, there are some operational characteristics that might be of interest to administrators:

- The maximum number of groups to which a GS client can subscribe or that a GS client can join is equivalent to the largest value containable in a signed integer variable.
- The maximum number of groups allowed within a domain is 65,535.
- These limits are the theoretical maximum limits. In practice, the amount of memory available to the Group Services daemon and its clients will reduce the limits to smaller values.

Group Services procedures

For the most part the Group Services subsystem runs itself without requiring administrator intervention. However, on occasion, you may need to check the status of the subsystem.

Displaying the status of the Group Services daemon

You can display the operational status of the Group Services daemon by issuing the **lssrc** command, enter:

```
lssrc -l -s cthags
```

In response, the **lssrc** command writes the status information to standard output. The information includes:

- The information provided by the **lssrc -s cthags** command (short form)
- The number of currently connected clients and their process IDs
- The status of the Group Services domain
- The node number on which the GS nameserver is running
- Statistics for client groups with providers or subscribers on this node.

Note that if the **lssrc** command times out, the Group Services daemon is probably unable to connect to the Topology Services subsystem. For more information, see “Group Services initialization errors” on page 192.

This sample output is from the **lssrc -l -s cthags** command on a node in the cluster:

```
Subsystem      Group          PID      Status
cthags         cthags         11938    active
4 locally-connected clients. Their PIDs:
21344(sample_test1) 17000(sample_test3) 18200(rmcd)
HA Group Services domain information:
Domain established by node 9.
Number of groups known locally: 2
Group name      Number of      Number of local
                providers     providers/subscribers
WomSchg_1       5              1              1
rmc_peers       7              1              0
```

In this domain, the GS nameserver is on node 9 of the system.

If a GS nameserver has not yet been established, the status indicates that the domain is not established. Similarly, if the GS nameserver fails, the status shows that the domain is recovering. Both of these conditions should clear in a short time. If they do not and the Topology Services subsystem is active, call the IBM Support Center.

Appendix. Product-related information

Reliable Scalable Cluster Technology (RSCT) is a component of the following licensed programs:

- AIX 5L
- Cluster Systems Management (CSM) for Linux
- System Automation for Multiplatforms

RSCT version

This edition applies to RSCT version:

- 2.4.1.0 for AIX 5.3
- 2.3.5.0 for AIX 5.2
- 2.4.1.0 for Linux

To find out which version of RSCT is running on a particular AIX node, enter:

```
lspp -L rsct.basic.rte
```

To find out which version of RSCT is running on a particular Linux node, enter:

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

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with user interfaces. Consult the assistive technology documentation for specific information when using such products to access interfaces.

ISO 9000

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

Product-related feedback

To contact the IBM cluster development organization, send your comments by e-mail to: cluster@us.ibm.com

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.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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:

For AIX:
IBM Corporation
Department LRAS, Building 003
11400 Burnet Road
Austin, Texas 78758-3498
U.S.A.

For Linux:
IBM Corporation
Department LJEB, MS P905
2455 South 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 illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. 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: The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX
AIX 5L
eServer
HACMP
IBM
IBM(logo)
IBMLink
iSeries
LoadLeveler
PowerPC
pSeries
RS/6000
SP
Tivoli
xSeries
zSeries

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

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

Glossary

access control. The process of limiting access to system objects and resources to authorized principals.

access control list. A list of principals and the type of access allowed to each.

ACL. See *access control list*.

action. The part of the event response resource that contains a command and other information about the command.

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

AIX. Advanced Interactive Executive. See *AIX operating system*.

AIX operating system. IBM's implementation of the UNIX operating system.

authentication. The process of validating the identity of an entity, generally based on user name and password. However, it does not address the access rights of that entity. Thus, it simply makes sure a user is who he or she claims to be.

authorization. The process of granting or denying access to an entity to system objects or resources, based on the entity's identity.

client. Client applications are the ordinary user interface programs that are invoked by users or routines provided by trusted services for other components to use. The client has no network identity of its own: it assumes the identity of the invoking user or of the process where it is called, who must have previously obtained network credentials.

cluster. A group of servers and other resources that act like a single system and enable high availability and, in some cases, load balancing and parallel processing.

clustering. The use of multiple computers (such as UNIX workstations, for example), multiple storage devices, and redundant interconnections to form what appears to users as a single highly-available system. Clustering can be used for load balancing, for high availability, and as a relatively low-cost form of parallel processing for scientific and other applications that lend themselves to parallel operations.

cluster security services. A component of RSCT that is used by RSCT applications and other RSCT components to perform authentication within both management domains and peer domains.

condition. A state of a resource as defined by the event response resource manager (ERRM) that is of interest to a client. It is defined by means of a logical expression called an event expression. Conditions apply to resource classes unless a specific resource is designated.

condition/response association. A link between a condition and a response.

CSM. Clusters Systems Management.

domain. (1) A set of network resources (such as applications and printers, for example) for a group of users. A user logs in to the domain to gain access to the resources, which could be located on a number of different servers in the network. (2) A group of server and client machines that exist in the same security structure. (3) A group of computers and devices on a network that are administered as a unit with common rules and procedures. Within the Internet, a domain is defined by its Internet Protocol (IP) address. All devices that share a common part of the IP address are said to be in the same domain.

event. Occurs when the event expression of a condition evaluates to True. An evaluation occurs each time an instance of a dynamic attribute is observed.

event expression. A definition of the specific state when an event is true.

event response. One or more actions as defined by the event response resource manager (ERRM) that take place in response to an event or a rearm event.

| **failover.** A backup operation that automatically
| switches to another adapter if one adapter fails. Failover
| is an important fault-tolerance function of mission-critical
| systems that rely on constant accessibility. Automatically
| and transparently to the user, failover redirects requests
| from the failed adapter to another adapter that mimics
| the operations of the failed adapter.

FFDC. See *first failure data capture*.

first failure data capture. Provides a way to track problems back to their origin even though the source problem may have occurred in other layers or subsystems than the layer or subsystem with which the end user is interacting. FFDC provides a correlator called an **ffdc_id** for any error that it writes to the AIX error log. This correlator can be used to link related events together to form a chain.

FIFO. First in first out, usually referring to buffers.

LAPI. See *low-level application programming interface*.

Linux. A freeware clone of UNIX for 386-based personal computers (PCs). Linux consists of the **linux** kernel (core operating system), originally written by Linus Torvalds, along with utility programs developed by the Free Software Foundation and by others.

low-level application programming interface. A low-level (low overhead) message-passing protocol that uses a one-sided communication model and active message paradigm to transfer data among tasks. See also *RSCT LAPI*. Contrast with *PSSP LAPI*.

logical unit number. A unique identifier used on a SCSI bus that enables it to differentiate between up to eight separate devices (each of which is a logical unit). Each LUN is a unique number that identifies a specific logical unit, which may be an end user, a file, or an application program.

LUN. See *logical unit number*.

management domain. A set of nodes configured for manageability by the Clusters Systems Management (CSM) licensed program. Such a domain has a management server that is used to administer a number of managed nodes. Only management servers have knowledge of the whole domain. Managed nodes only know about the servers managing them; they know nothing of each other. Contrast with *peer domain*.

mutex. See *mutual exclusion object*.

mutual exclusion object. A program object that allows multiple program threads to share the same resource, such as file access, but not simultaneously. When a program is started, a mutual exclusion object is created with a unique name. After this stage, any thread that needs the resource must lock the mutual exclusion object from other threads while it is using the resource. The mutual exclusion object is set to unlock when the data is no longer needed or the routine is finished.

network credentials. These represent the data specific to each underlying security mechanism.

OSI. Operating system image.

PAC. See *privileged attribute certificate*.

Parallel System Support Programs. The IBM Parallel System Support Programs for AIX 5L (PSSP) licensed program is system administration software for the IBM RS/6000® SP system.

peer domain. A set of nodes configured for high availability by the configuration resource manager. Such a domain has no distinguished or master node. All nodes are aware of all other nodes, and administrative commands can be issued from any node in the domain.

All nodes also have a consistent view of the domain membership. Contrast with *management domain*.

principal. A user, an instance of the server, or an instance of a trusted client whose identity is to be authenticated.

privileged attribute certificate. Contains such information as the client's name and the groups to which it belongs. Its format is dependent on the underlying security mechanism.

rearm event. Occurs when the rearm expression for a condition evaluates to True.

rearm expression. An expression that generates an event which alternates with an original event in the following way: the event expression is used until it is true; then, the rearm expression is used until it is true; then, the event expression is used. The rearm expression is commonly the inverse of the event expression. It can also be used with the event expression to define an upper and lower boundary for a condition of interest.

PSSP. See *Parallel System Support Programs*.

PSSP LAPI. The version of LAPI that takes advantage of the SP Switch.

Reliable Scalable Cluster Technology. A set of software components that together provide a comprehensive clustering environment for AIX and Linux. RSCT is the infrastructure used by a variety of IBM products to provide clusters with improved system availability, scalability, and ease of use.

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

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

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

RSCT. See *Reliable Scalable Cluster Technology*.

RSCT LAPI. The version of LAPI that takes advantage of the IBM @server pSeries High Performance Switch (pSeries HPS). See also *low-level application programming interface*.

RSCT peer domain. See *peer domain*.

SCSI. See *Small System Computer Interface*.

Small System Computer Interface. A parallel interface that can have up to eight devices all attached through a single cable; the cable and the host (computer) adapter make up the SCSI bus. The bus allows the interchange of information between devices independently of the host. In the SCSI program, each device is assigned a unique number, which is either a number between 0 and 7 for an 8-bit (narrow) bus, or between 8 and 16 for a 16-bit (wide) bus. The devices that request input/output (I/O) operations are initiators and the devices that perform these operations are targets. Each target has the capacity to connect up to eight additional devices through its own controller; these devices are the logical units, each of which is assigned a unique number for identification to the SCSI controller for command processing.

SD. Structured data.

security context token. A pointer to an opaque data structure called the context token descriptor. The context token is associated with a connection between a client and the server.

security services token. A pointer to an opaque descriptor called the security token descriptor. It keeps track of the mechanism-independent information and state.

servers. Server programs are usually daemons or other applications running in the background without a user's inherited credentials. A server must acquire its own network identity to get to access other trusted services.

standalone system. A system on which you are using LAPI that is not running IBM's Parallel Environment for AIX (PE) licensed program.

TCP. See *transmission control protocol*.

transmission control protocol. One of the core Internet protocols. TCP ports are 16-bit entities, so that a maximum of 65535 different endpoints are possible within a single IP address.

UDP. See *user datagram protocol*.

user datagram protocol. One of the core Internet protocols. It is a layer 4 protocol (Transport layer of the OSI model) within the Internet protocol suite. It provides a mechanism to identify different endpoints on a single host by means of ports. UDP deals with single packet delivery, provided by the underlying IP. As a stateless

protocol, it is often used in such applications where data must arrive quickly. The benefit of this smaller feature set is quicker data transmittal and lower total overhead. UDP packets (also known as datagrams) contain, in addition to the lower-level headers, a UDP header, which consists of a checksum, the packet length, plus source, and destination ports. As with TCP, UDP ports are 16-bit entities, so that a maximum of 65535 different endpoints are possible within a single IP address.

Index

Special characters

- /etc/services file
 - use by Group Services 187
- /var file system
 - and Group Services tracing 190

A

- accessibility 195
- adding subsystems
 - Group Services (cthagsctrl) 190
- addrpnode command 34
- audience of this book vii

B

- base data types, supported 132
- bibliography ix
- blanks, use of in expressions 135
- books
 - RSCT ix

C

- CIM (Common Information Model) 122
- CIM Resource Manager 122
- cleaning subsystems
 - Group Services (cthagsctrl) 190
- client communication
 - with Group Services subsystem 186
- client, Group Services
 - definition 185
- commands
 - cthats 171
 - cthatstune 171
- Common Information Model (CIM) 122
- communication groups (in an RSCT peer domain)
 - creating 44
 - listing 40
 - modifying 41
 - removing 46
 - started automatically when peer domain is brought online 29
- communication, client
 - with Group Services subsystem 186
- communications, Group Services
 - between Group Services daemons 187
 - local GS clients 188
- configuration resource manager 19
- configuration resource manager commands
 - addrpnode 34
 - lscomg 40
 - mkcomg 44
 - mkrpdomain 28
 - preprnode 26, 32
 - rmcomg 46
 - rmpdomain 38

- configuration resource manager commands (*continued*)
 - rmpnode 37
 - startprdomain 29, 35
 - startprnode 35
 - stopprdomain 37
 - stopprnode 36
- core file
 - Group Services daemon 189
- cthagsctrl command
 - adding the Group Services subsystem 190
 - cleaning the Group Services subsystem 190
 - control command for Group Services 188
 - deleting the Group Services subsystem 190
 - starting the Group Services subsystem 190
 - stopping the Group Services subsystem 190
 - summary of functions 189
 - tracing the Group Services subsystem 190
- cthats command 171
- cthatstune command 171

D

- data types used for literal values 133
- data types, base 132
- data types, structured 133
- definitions 201
- deleting subsystems
 - Group Services (cthagsctrl) 190
- disability 195
- domain, operational
 - for Group Services 186

E

- expressions
 - pattern matching supported in 139
- expressions, operators for 135

F

- feedback
 - product-related 195
- files and directories
 - component of Group Services 188

G

- glossary 201
- group membership list
 - definition 185
- group services
 - started by the configuration resource manager in an RSCT peer domain 29
- Group Services
 - abnormal termination of cthagsctrl add 190
 - disk space and tracing 190
 - performance and tracing 190

- Group Services API (GSAPI)
 - component of Group Services 187
- Group Services client
 - definition 185
- Group Services communications
 - between Group Services daemons 187
 - local GS clients 188
- Group Services daemon
 - abnormal termination core file 189
 - communications 187
 - component of Group Services 186
 - cthagsctrl control command 188
 - getting status 192, 193
 - initialization 190
 - initialization errors 192
 - operation 192
 - recovery from failure (automatic) 192
 - trace output log file 188
- Group Services subsystem
 - adding with cthagsctrl command 190
 - cleaning with cthagsctrl command 190
 - client communication 186
 - component summary 186
 - components 186, 189
 - configuration 189
 - configuring and operating 185, 193
 - deleting with cthagsctrl command 190
 - dependencies 189
 - getting subsystem status 192, 193
 - Group Services daemon initialization 190
 - Group Services daemon initialization errors 192
 - Group Services daemon operation 192
 - initialization errors 192
 - introducing 185
 - operational domain 186
 - recovery from failure (automatic) 192
 - starting with cthagsctrl command 190
 - stopping with cthagsctrl command 190
 - tracing with cthagsctrl command 190
- group state value
 - definition 185
- group, Group Services
 - definition 185
- groups
 - Group Services
 - restrictions on number per client 192
 - restrictions on number per domain 192
- GS nameserver
 - establishing 191
- GSAPI (Group Services Application Programming Interface)
 - component of Group Services 187
- GSAPI libraries
 - location 187

H

- hagsd daemon
 - location 186
- high availability services
 - Group Services subsystem 185

I

- IBM.LPCCommands resource class
 - usage 141
- ISO 9000 195

L

- least-privilege (LP) resource manager
 - using for access control to root commands or scripts 141
 - using for remote execution of root commands or scripts 141
- local GS clients
 - Group Services communications 188
- lock file
 - Group Services 188
- log file
 - Group Services 188
- LookAt x
- lscomg command 40
- lssrc command
 - getting Group Services status 192

M

- mkcomg command 44
- mkrpdomain command 28

N

- nameserver, Group Services
 - establishing 191
- Network Interface Modules (NIM) 45
- NIM 45

O

- operator precedence 137
- operators available for use in expressions 135

P

- pattern matching supported in expressions 139
- port numbers
 - component of Group Services 187
 - topology services 170
- port numbers, specifying for Topology Services and Group Services in configuration resource manager 27
- precedence of operators 137
- preprnode command 26, 32
- prerequisite information ix
- prerequisite knowledge for this book vii
- problem determination
 - Group Services subsystem
 - abnormal termination core file 189
 - abnormal termination of cthagsctrl add 190
 - getting subsystem status 192
 - tracing 190
- product-related feedback 195

- protocol, Group Services
 - definition 185
- provider
 - definition 185
- publications
 - RSCT ix

R

- recovery from failure
 - Group Services 192
- related information ix
- resource class
 - IBM.LPCommands 141
- restrictions
 - Group Services
 - groups per client 192
 - groups per domain 192
- rmcomg command 46
- rmrpdomain command 38
- rmrpnode command 37
- root command
 - controlling access through least-privilege resource manager 141
 - executing on local or remote node 141
- RSCT
 - books ix
 - feedback 195
 - publications ix
- RSCT peer domain
 - adding a node to a 34
 - bringing a node online in a 35
 - bringing online 29
 - creating 27
 - removing a node from a 37
 - removing a peer domain 38
 - security environment, preparing 26, 32
 - taking a peer domain node offline 36
 - taking peer domain offline 37
- RSCT version 195

S

- script
 - controlling access through least-privilege resource manager 141
 - executing on local or remote node 141
- SDR (System Data Repository)
 - and cthagsctrl clean 190
- security
 - controlling access to root commands or scripts 141
 - preparing security environment for an RSCT peer domain 26, 32
- sockets
 - component of Group Services 187
 - topology services 170
- SRC (System Resource Controller)
 - and Group Services daemon 191
 - dependency by Group Services 189
- starting
 - Topology Services 171

- starting subsystems
 - Group Services (cthagsctrl) 190
- startprdomain command 29, 35
- startprnode command 35
- status, Group Services
 - output of lssrc command 192, 193
- stopping subsystems
 - Group Services (cthagsctrl) 190
- stopprdomain command 37
- stopprnode command 36
- structured data types 133
- subscriber
 - definition 185
- subsystem
 - Group Services 185, 193
 - Topology Services 167
- subsystem status
 - for Group Services 192, 193
- System Data Repository (SDR)
 - and cthagsctrl clean 190
- System Resource Controller (SRC)
 - and Group Services daemon 191
 - dependency by Group Services 189

T

- tasks
 - changing an LP resource
 - steps for 145
 - controlling access to root commands and scripts
 - roadmap 141
 - defining LP resources
 - steps for 143
 - defining LPRM authorized users
 - steps for 143
 - removing LP resources
 - steps for 145
 - running an LP resource
 - step for 144
- terminology 201
- time limits
 - Group Services
 - connection to Topology Services 191
- topology services
 - communicating 170
 - components 168
 - configuring 175
 - control 171
 - daemon 168
 - defaults 178
 - dependencies 173
 - directories 172
 - files 172
 - initializing 176
 - introducing 167
 - limitations 178
 - operating 177
 - port numbers 170
 - procedures 182
 - refreshing 181
 - sockets 170

- topology services (*continued*)
 - started by the configuration resource manager in an RSCT peer domain 29
 - status 182
 - tuning 179
- Topology Services subsystem
 - and Group Services daemon initialization 191
 - configuring and operating 167
 - dependency by Group Services 189
- trace output log
 - Group Services 188
- tracing subsystems
 - Group Services (cthagsctrl) 190
- trademarks 199
- troubleshooting
 - Group Services subsystem
 - abnormal termination core file 189
 - abnormal termination of cthagsctrl add 190
 - getting subsystem status 192
 - initialization errors 192
 - tracing 190
- tuning
 - Topology Services 171

U

- UDP port
 - use by Group Services 187
- UNIX domain socket
 - Group Services client communication 186
 - use by Group Services 187

V

- variable names 135
- variable names, restrictions for 135
- version
 - of RSCT 195

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

IBM Reliable Scalable Cluster Technology
Administration Guide

Publication No. SA22-7889-06

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

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>				

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

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>				
Complete	<input type="checkbox"/>				
Easy to find	<input type="checkbox"/>				
Easy to understand	<input type="checkbox"/>				
Well organized	<input type="checkbox"/>				
Applicable to your tasks	<input type="checkbox"/>				

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.



Fold and Tape

Please do not staple

Fold and Tape



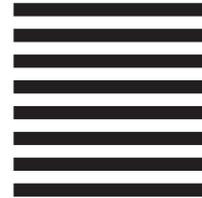
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

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



Fold and Tape

Please do not staple

Fold and Tape



Program Number: 5765-E62, 5765-G03, 5765-G16, 5765-E88, 5639-N53,
5655-I53

SA22-7889-06

